

AD-A141 198

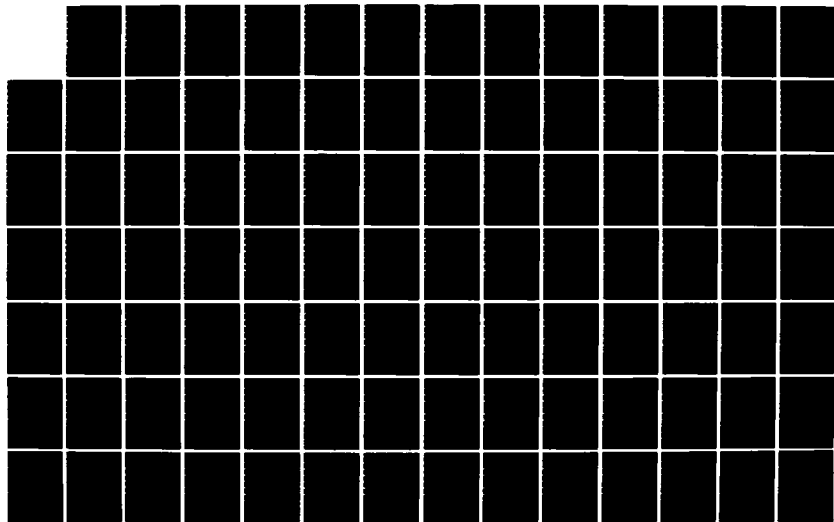
ENHANCEMENTS TO THE NETWORK REPAIR LEVEL ANALYSIS
(NRLA) MODEL USING MARG. (U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.

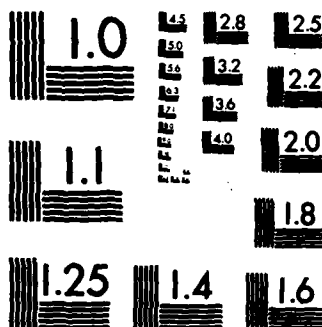
1/2

UNCLASSIFIED

G W ARNETT ET AL. DEC 83 AFIT/GOR/OS/83D-3 F/G 15/5

NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AFIT/GOR/OS/83D-3

AD-A141 198

ENHANCEMENTS TO THE NETWORK REPAIR
LEVEL ANALYSIS (NRLA) MODEL USING
MARGINAL ANALYSIS TECHNIQUES AND
CENTRALIZED INTERMEDIATE REPAIR
FACILITY (CIRF) MAINTENANCE CONCEPTS

THESIS

Gary W. Arnett
Captain, USAF

Nicholas W. Reybrock
Captain, USAF

AFIT/GOR/OS/83D-3

DTIC FILE COPY

DTIC
SELECTE

15 034

Approved for public release: IAW AFR 100-17.

LYNN E. WOLAVER
Dean for Research and Professional Development
Air Force Institute of Technology
Wright-Patterson AFB OH 45433

84 05 15 034

AFIT/GOR/OS/83D-3

ENHANCEMENTS TO THE NETWORK REPAIR LEVEL ANALYSIS (NRLA)
MODEL USING MARGINAL ANALYSIS TECHNIQUES AND
CENTRALIZED INTERMEDIATE REPAIR FACILITY (CIRF)
MAINTENANCE CONCEPTS

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University

In Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Operations Research

Gary W. Arnett
Captain, USAF

Nicholas W. Reybrock
Captain, USAF

December 1983

Approved for public release; distribution unlimited

Acknowledgements

This research effort has been conducted under the direct sponsorship of the Concepts and Analysis Division, Air Force Acquisition Logistics Center (XRS/AFLC) Wright Patterson Air Force Base, Ohio. Mr. Larry Brisken provided us the initial guidance and understanding of the NRLA model, which helped us define our objectives and develop a thesis strategy.

Mr. Terry Mitchell's perceptive insights and detailed knowledge of the program's logic have been of immeasurable value to the successful accomplishment of our intended objectives. We also especially express our appreciation and thanks to our advisor, Lt Col Gerald R. Armstrong, and our reader, Lt Col Tom Clark, Jr., for their professional guidance and technical expertise.

Finally, we take this opportunity to thank our wives, Kay and Michelle, and our children, Becky, Brian, and Kasey, whose support and help made the completion of this product a reality.

Gary W. Arnett

Nicholas W. Reybrock



Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Date	
A	
JF	
11/11	

A-1

Table of Contents

	Page
Acknowledgements	ii
List of Figures	v
List of Tables	vi
Abstract	vii
I. Problem Definition	1
Introduction	1
Background	1
NRLA	4
Assumptions	5
Limitations	9
Thesis Objectives	11
Organization	15
II. Literature Review	17
Introduction	17
Repair Level Analysis- A Macro View	17
Max-flo Min-cut Problem	23
Mathematical Formulation	24
Max-flo Problem	25
Min-cut Problem	25
Labeling Algorithms	26
Breadth First Algorithm	28
Depth First Algorithm	30
Minimum Cut Algorithm	32
Generalized Network Primal Algorithm	34
III. Methodology	36
IV. Marginal Analysis Enhancement	42
Marginal Repair Level Analysis (MARLA)	42
Concept Development	42
Computerization/Integration	47
Verification/Validation	52
Analysis/Results	54
V. Max-Flo Min-Cut Labeling Enhancement	56
NRLA Network Formulation	56
Network Structure	56
Minimum Cuts	58

Max-flo Min-cut Algorithmic Improvements	61
Concept Development	61
Breadth First Labeling Algorithmic.....	62
Depth First Labeling Algorithmic.....	67
Computerization/Integration	69
Verification/Validation	70
Analysis/Results	70
VI. Centralized Intermediate Repair Enhancement	73
Concept Development.....	73
Input Data Requirements.....	74
Cost Requirements.....	76
CIRF Network Conceptual Development.....	77
CIRF Network Construction.....	78
CIRF Network Cut Sets.....	83
Computerization/Integration.....	87
Verification/Validation.....	87
Analysis/Results	89
VII. Conclusions and Recommendations	90
General Comments	90
Recommendations	91
MARLA	91
Labeling Improvements	91
CIRF	92
Conclusion	92
Appendix A: Glossary of CIRF and MARLA	
Variables and Arrays	94
Appendix B: NRLA Subroutine Descriptions	97
Appendix C: CRFSET Subroutine	99
Appendix D: MARLA Subroutine	106
Appendix E: MSETNT Subroutine	113
Appendix F: NMXFLO Subroutine	121
Appendix G: OUTPUT Subroutine	126
Appendix H: SECMP Subroutine	135
Bibliography	139
Vita	141

List of Figures

Figure	Page
1. Basic Structure of a RLA Network	7
2. Centralized Intermediate Repair Facility (CIRF) Structure	15
3. Reparable Assets Structure	19
4. Shared-Fixed Cost Network	21
5. Example Max-Flow Min-Cut Network	24
6. NRLA Program Logic	37
7. Thesis Task Organization	38
8. NRLA Thesis Methodology	40
9. Example of Proposed Arc Elimination Process	48
10. MARLA SE Requirements Logic	51
11. Potential Cuts of the RLA Network	59
12. Network Labeling Example.....	65
13a. CIRF RLA Subnetworks	79
13g.	80
14a. CIRF RLA Network Potential Cuts	84
14g.	85

List of Tables

Table	Page
I. Relationship Between Logistics and Network Costs	8
II. Repair Level Costs	43
III. Marginal Analysis Decision Criteria	46
IV. RLA Network Decision Criteria	60
V. Max-Flow Algorithm Test Results	71
VI. CIRF Network Decision Costs	86

Abstract

→ Applied network theory and marginal analysis concepts were utilized to design, computerize, verify, and evaluate, three major software modifications to the Network Repair Level Analysis (NRLA) model. First, a preprocessor subroutine using marginal analysis techniques was developed and tested to reduce the computer processing requirements of the program. Second, a new network labeling algorithm which solve the max-flow min-cut problem is presented. This algorithm performs 100 times faster than the current algorithm and 73 times faster than the highly efficient, commercially available, primal networking code known as GNET. Third, for the first time a networking structure has been designed which allows for the inclusion of Centralized Intermediate Repair Facilities (CIRF) in the repair level analysis decision process.

These products greatly expand the NRLA model's capability while at the same time improving its operational efficiency. Through their integration and use, System Program Managers have a comprehensive analytical tool to effectively conduct repair level analysis and to design more cost-effective logistical structures to support the operation of Air Force systems.

→ The NRLA program is hosted on the CREATE Operating System and contains approximately 5500 lines of computer code. It consists of a main routine and twelve major subroutines. The

results from the NRLA model are used by logistical planners to quantify the potential cost impacts associated with alternative maintenance plans. As the technological complexity of weapons systems has increased new and innovative logistical support systems are required to maximize the system's operational capability while minimizing life cycle costs. The above enhancements to the NRLA model were designed to meet these new challenges. This research effort was sponsored by the Concepts and Analysis Division, Air Force Acquisition Logistics Center (XRS/AFALC/AFLC).

I. Problem Definition

Introduction

Every program manager, who is responsible for acquiring a new weapon system to be used by tomorrow's United States Air Force (USAF), faces a common challenge. It consists of three, sometimes conflicting, functional goals; system performance, system cost, and acquisition schedule. His goal is a formidable one. He must attempt to maximize the performance of the system at the least possible cost to the government while maintaining the scheduled time-frame for procurement of the system. A primary ingredient in the successful accomplishment of this effort is an accurate assessment of the "operational support requirements and limitations of the system" (13:3) and the timely consideration of alternative maintenance concepts capable of achieving a desired level of system effectiveness.

During the Vietnam conflict, the importance of developing a valid maintenance plan as an integral part of the system's engineering development became brutally apparent. Many times new systems were deployed to meet the changing threat only to have their components subsequently fail causing excessive downtime for the system. An example of this was the AN/TRC-87 UHF radio set. The design of this radio set did not anticipate its extensive use in a jungle environment, as a result major component failures occurred

which required depot repair (1:134). The availability, dependability, and capability of these systems were all substantially reduced because of a lack of adequate attention being given to their reliability, the resources necessary to support their repair, and specific identification of the repair location for their components. As a direct result of these experiences, the Department of Defense (DOD) community and the Air Force have taken great strides to ensure that such decisions become an integral part of the system's design. The Integrated Logistic Support (ILS) Program, established in 1972 by AFR 800-8, requires that planning for cost-effective logistic support, including analysis of repair-level alternatives, be conducted during the acquisition phase of a weapon system (11:3). In conjunction with this guidance, "AFSC/AFLCR 800-28 establishes Air Force policies and procedures with respect to Repair Level Analysis (RLA)" (14:4). These policies and procedures are designed to ensure that alternative maintenance concepts are considered during conceptual development and at the appropriate times in the life cycle of the system. Additionally, AFSC/AFLC Regulation 800-28 identifies RLA as a separate evaluation factor during the source selection process. This is a further indication of the emphasis and recognition that is now being given repair level decision making (13:3).

The Repair Level Analysis process is the procedure by which economic comparisons are made between repair locations to develop a comprehensive and cost effective maintenance plan. RLA encompasses a wide variety of analytical techni-

ques and methods. These techniques can be used separately or in combination to provide the system program offices with economically based management information as to the "best" set of repair level decisions for their system. In an effort to document the strengths and weaknesses of these different methods and to assist logistics analysts in determining the appropriate technique for a particular application, the Air Force has recently updated these procedures in AFLC/AFSC Pamphlet 800-4, Repair Level Analysis Procedures (revised June 1983). Formerly known as Optimum Repair Level Analysis (ORLA) (12), RLA now encompasses the following methods (13:5):

- (1) Network Repair Level Analysis (NRLA)
- (2) Item Repair Level Analysis (IRLA)
- (3) Marginal Analysis Repair Level Analysis (MARLA)
- (4) Equal Cost Curves (ECC)
- (5) SE/Pipeline Ratio

Each of these techniques are clearly explained in AFLCP/AFSCP 800-4. However, the NRLA conceptual model, which were introduced in 1980, are recognized as the most comprehensive approach to repair level analysis and has consequently received the widest use and attention (13:5). Since NRLA's implementation, a question has arisen as to its ability to continue to provide reliable and consistent repair level logistics cost effectiveness and the operational capability of future systems. This research has been initiated in an effort to answer this question. The remainder of this chapter will deal with the scope and limitations of the NRLA model with an emphasis on identifying software enhancements to correct identified shortcomings.

Background

Prior to 1980 repair level analysis for Air Force systems was conducted on an item by item basis using a method called Optimum Repair Level Analysis (ORLA) (12:1). In ORLA each Line Replaceable Unit (LRU) and Shop Replaceable Unit (SRU) in the system would be separately priced for each of three repair options; (1) repair at intermediate base, (2) repair at depot and (3) discard or scrap. The repair option associated with the minimum cost would be selected for that particular LRU or SRU. The problem with this method of determining repair level decisions was that the SE costs associated with each LRU/SRU repair were prorated and estimated according to that LRU/SRU's use of the SE resource. Thus a LRU/SRU which required 20 percent of the SE's available repair time would need to economically justify at least 20 percent of that SE cost. In addition if all LRU/SRUs requiring that SE were not chosen for the same repair level then the percent utilization and prorated costs would need to be recomputed based on the new set of LRU/SRUs at that level. This could then lead to other LRU/SRUs not being able to support their required SE costs which would lead to further proration of costs and possibly to all LRU/SRUs being repaired at the depot level when in fact the total set of SE related LRU/SRUs could easily justify base level repair as the most economical option (8:6-7). A second problem with the ORLA method of determining repair level decisions was that the Air force had no official computer

program to implement ORLA. This resulted in virtual chaos when evaluating contractor estimates, in that contractors did not use standardized software to conduct their analysis.

To solve these problems the NRLA program was developed by the Concepts and Analysis division, XRS/AFALC(AFLC) giving the Air force, for the first time, the capability to use a systems approach in the development and formulation of repair level decisions. NRLA was developed as a FORTRAN based software package that is transportable between computer systems and is therefore useable by government contractors providing a consistent framework for evaluation of their proposals.

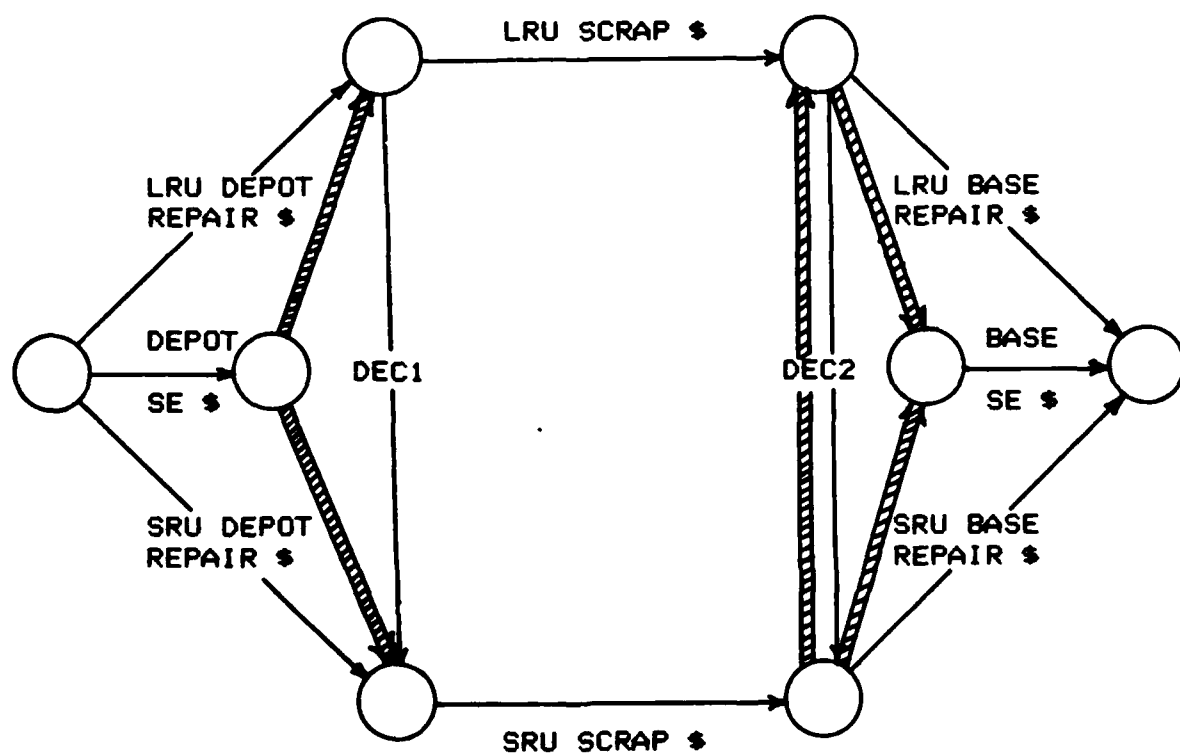
NRLA

The formulation of the repair level analysis problem as a network provides a couple of advantages over previous RLA methodologies. It takes into account the LRU and SRU indenture relationships which keeps it from making inconsistent decisions. An example of this would be deciding to scrap a LRU but repair its indentured SRUs. The model also treats each piece of support equipment as a common resource which is shared by a group of LRUs and/or SRUs. The repair level decisions are then made, based on these LRU/SRU group's ability to economically support the purchase of the support equipment for a particular repair level. All of the repair decisions are determined simultaneously for all of the failure modes of a group of LRUs and their associated SRUs, thus the decisions that are made are optimal for the

entire group of items.

The actual formulation of the network is shown in Figure 1. The nodes of the network represent the LRU failure modes, the SRUs, and the SE resources required for the LRU/SRU repairs. The arcs of the network represent the costs associated with the different repair level options. Table I defines the specific arc costs for the network in terms of eleven types of logistical costs. The NRLA Users Guide available from XRS/AFALC(AFLC) details the computation of these costs in terms of the LRU/SRU's cost, MTBF, and standard maintenance and supply factors. The heavy arcs are dummy arcs which provide the LRU/SRU/SE interdependency relationships. The capacities of the dummy arcs are set such that they will never enter into the solution set.

The repair level decisions are obtained by applying an optimization algorithm to the network to solve the maximin flow minimum cut problem. This algorithm provides a network solution with a cut set that describes the unique optimum set of minimum cost repair level decisions for the entire system. It should be pointed out that the NRLA model in no way attempts to compute a total life cycle cost for a particular set of repair decisions but, "includes only those costs which directly impact the repair level decision". Examples of costs which are not included are: repair in place costs, and the costs associated with removing the failed LRU/SRU from the end item (8:3).



NOTE: DEC1 and DEC2 are SRU costs which are incurred if the LRU decision is base level repair and (a) the SRU decision is depot repair (DEC1), or (b) the SRU decision is either depot repair or scrap (DEC2).

FIGURE 1. Basic Structure of an RLA Network

TABLE I

Relationship Between Logistics and Network Costs

NETWORK DECISION FACTORS										
	LRU REPAIR			SRU REPAIR					SE	
	D E P O T	S C R A P	B A S E	D E P O T	S C R A P	B A S E	D E C 1	D E C 2	D E P O T	B A S E
LOGISTIC FACTORS										
1. Support Equipment									X	X
(1a) Acquisition									x	x
(1b) Ops & Maint									x	x
(1c) Facilities									x	x
2. Tech. Data Acquis.	X		X	X		X				
3. Maint Training	X		X	X		X				
4. Repair Labor	X		X	X		X				
5. Item Entry	X		X	X		X				
6. Supply Admin.			X			X				
7. Repair Material	X		X	X		X				
8. Packing & Shipping	X	X	X			X	X	X		
9. Base Spares Quantity	X	X	X			X		X		
10. Depot Spares Quantity	X						X			
11. Replacement Spares		X			X					

Assumptions

Virtually any model makes some simplifying assumptions which enables it to calculate a solution or make some determination/conclusion about the system being studied. The NRLA model is no exception to this rule. The following assumptions have been identified as relevant to the solutions derived by NRLA (8:3-6).

- (1) The user specifies the number of bases and the number of end items per base (assumed to be equal for all bases). All depot repair of a particular LRU/SRU is accomplished at the same depot location.
- (2) Base level maintenance system data are equal for all bases and all types of repair tasks. The corresponding depot data factors are constant also.
- (3) Supply system data factors are constant for all LRUs and SRUs analyzed. ie. shipping times from depot to any CONUS location are equal, and the same is true for shipments from depot to any overseas location.
- (4) Only a single set of technical data is purchased from the contractor.
- (5) Preventative and scheduled maintenance actions are not addressed by the model.
- (6) The model explicitly evaluates each LRU failure mode for a repair level decision, however SRU failure modes are assumed to be similar enough to allow for considering them all equal.
- (7) Maintenance man-hours for repair and SE utilization are assumed to be equal.
- (8) The depot stock level of SRUs is designed to satisfy base level demands. The stock level supports base level SRU remove and replace maintenance actions but not similar depot actions.

Limitations

The NRLA program as it exists today has proven to be a valuable tool in such areas as determining optimal repair

levels in support of the provisioning process, identification and justification of LRU/SRU support equipment requirements, and assessment of a system's sensitivity to cost and reliability growth. However, limitations in the programs efficiency and capability have resulted in the need to enhance and in some cases expand the software of the program to accommodate larger and more complex systems and to be able to evaluate new maintenance concepts.

Approximately 50-60 contractors and system program offices are currently using the NRLA model as implemented in 1980. User comments indicate that approximately 60% of the analyses that they perform using NRLA are conducted on a piecemeal basis with subsystems being analyzed individually. This is due to the fact that it is easier and more convenient to analyze and work with sub-systems. However, the main reason which drives users to analyze smaller sections of the system is computer resources. As the system being analyzed gets larger and more complex it requires more computer storage space to load the program and more computer time to run it. Both storage requirements and run-time requirements are used to prioritize computer jobs with the result that as the job gets bigger the turn around time on the analysis gets longer, an undesirable result. Also once the analysis of the subsystems is complete these results are then manually cross-referenced to determine the final systems set of repair level decisions.

A second limitation in the NRLA model results from the

evolution of a new maintenance concept, Centralized Intermediate Repair Facilities (CIRF). This concept is a result of the higher LRU/SRU/SE costs associated with new systems. In many instances the SE costs are too high to allow for base level repair and at the same time the pipeline costs associated with depot repair are also much higher than is desired. CIRF provides for a single central repair facility servicing several bases thereby eliminating some of the costs associated with each of the above repair options. The NRLA model currently does not allow for consideration of a CIRF maintenance concept in its optimum repair level analysis.

Thesis Objectives

The main objectives of the thesis effort deal with developing enhancements to the NRLA model which will provide solutions to the shortcomings as identified by the users; ie

- (1) The excessive computer time necessary to execute the program for large systems
- (2) Inability of the program to consider the option of a Centralized Intermediate Repair Facility (CIRF)

The objectives fall into two distinct areas; (1) enhancements which will allow the NRLA model to analyze larger and more complex systems and (2) enhancements which will allow the NRLA model to evaluate additional maintenance concepts, primarily CIRF. The first objective area can be further broken down into two areas (1) enhancements which increase the

efficiency of the algorithm which solves the max-flow min-cut problem and (2) enhancements which reduce the size of the problem input to the NRLA model. A short description of each of these objective areas and the proposed solution methodology follows. A comprehensive treatment of these areas is contained in Chapter III.

The computational time requirements for the solution of the repair level problem by the NRLA model are driven by two major factors, one internal to the program and one external. The internal factor deals with the computational efficiency of the labelling algorithm subroutine used to solve the max-flow min-cut problem. Labeling in the context of network flows is the means by which the algorithm identifies which nodes in the network it has visited. The labels prevent the algorithm from revisiting a node thus ensuring that the flow augmentation path will be identified in a finite number of steps. The external factor affecting the computation time is the size of the system to be analyzed in terms of LRUs, SRUs, and SE. Each of these areas is addressed in the thesis effort.

The current labeling algorithm utilized in the NRLA model is one developed by Ford and Fulkerson in the early 1960s (17). The approach taken is to perform a breadth first search with labeling as you proceed to the network sink. Once the sink is reached the flow is augmented on the path and the labels erased and the process started over. In a large system the majority of the processing time is used

performing the labeling process which can be very inefficient. The approach taken to reduce this processing time is to use a depth first search, as conceptually described by Horowitz and Sahni (19:268), for flow augmentation until the maximum flow is found. This is then followed by a breadth first search to provide the correct labels for the identification of the unique minimum cost set of repair level decisions.

In addition to the depth first approach for increasing the algorithm efficiency a second method of solving the problem using a large scale commercially available algorithm called GNET will be explored. This method is expected to provide a viable alternative for the solution of systems of extreme size, such as the B1 bomber and the MX missile. The GNET algorithm is a primal algorithm which has been used to solve extremely large transportation and transshipment problems (7). The NRLA program will be modified to include GNET as a subroutine to solve the max-flow problem, this will once again be followed by the Ford Fulkerson labeling process to identify the unique min cost solution needed by NRLA.

Creating a mechanism to control the second factor, that of system size, was initially much more difficult to conceptualize. Normally, the configuration of a proposed weapon system does not dramatically change once the preliminary design review (PDR) between the SPO and the contractor is conducted. Therefore the types and number of items (LRUs, SRUs, etc.) which make-up the total system are given as fixed

inputs to the program. However, a possible solution to this dilemma is the use of another RLA technique called Marginal Repair Level Analysis (MARLA). Simply stated, the approach is to reduce the number of items which have to be analyzed by NRLA. This will be accomplished by incorporating a marginal repair level analysis subroutine into the existing program. This software package, acting as a NRLA preprocessor, would calculate "marginal values for reparable, and then determine, based on those marginal values, whether an item should be scrapped, depot repaired, intermediate repaired "(13:67) or CIRF repaired.

The final aspect of NRLA which will be investigated is its ability to analyze a CIRF designed maintenance plan. The structure of the NRLA model is based on the traditional three-level maintenance concept (base, intermediate, depot) as detailed in AFR 60-5. Recently, an alternative maintenance approach known as Centralized Intermediate Repair Facilities (CIRF) has emerged as an efficient way to effectively support the repair of a weapons system. Normally each base has its own dedicated intermediate repair shop, which diagnose, repair, and replace system LRUs and SRUs. On the other hand, the CIRF concept utilizes a centrally located intermediate repair facility to service multiple bases (Figure 2). In order to ensure that a comprehensive economic analysis is performed when determining the optimal set of repair level decisions an expansion and modification of the program is necessary to evaluate the cost related impacts of using a

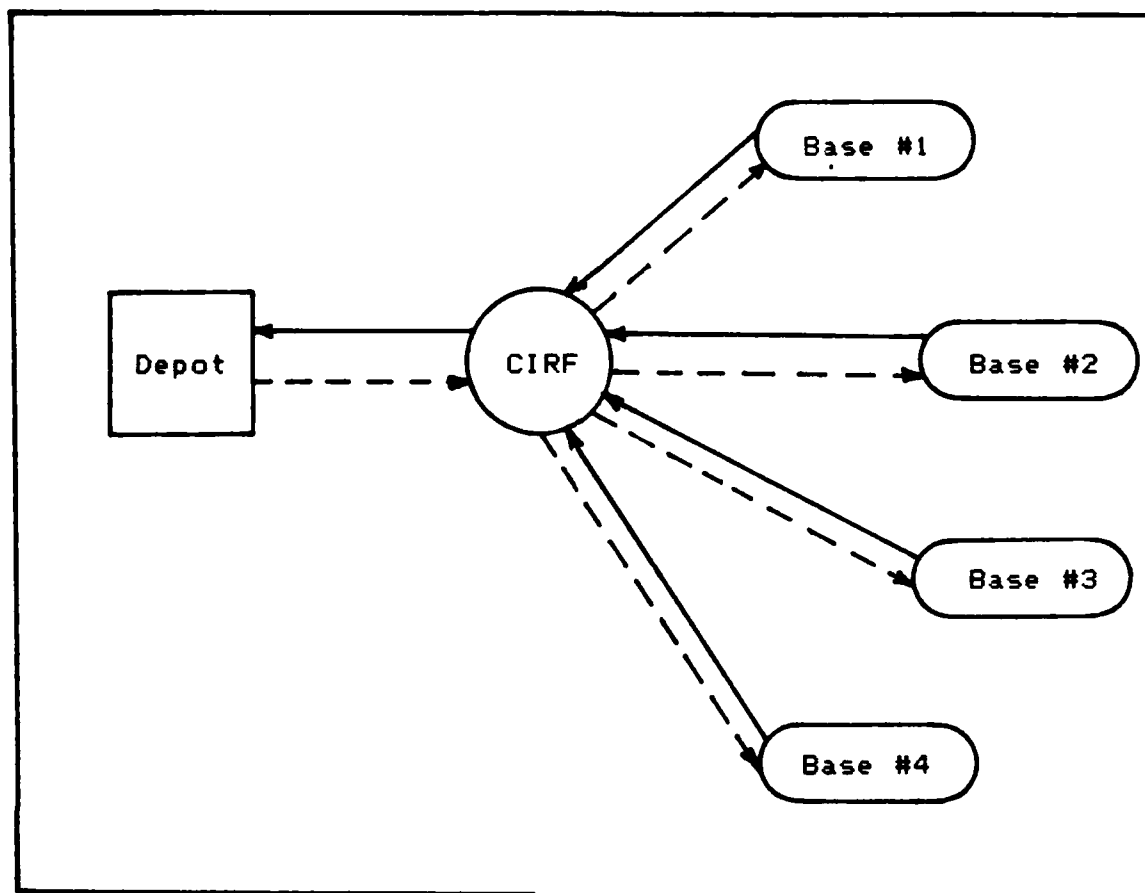


Figure 2. Centralized Intermediate Repair Facility Concept

CIRF maintenance approach. This will be the second of the two primary objectives of this thesis effort, the development of a network structure capable of analyzing a CIRF repair option. If possible this capability will be incorporated into the existing NRLA software program.

Organization

Chapter II encompasses a survey of the current literature available relating to networking algorithms with special

emphasis on those articles which formulate the repair level decision based on a network structure. The thesis methodology is presented in Chapter III with a detailed discussion of the structural model developed to accomplish the stated thesis objectives. An in-depth discussion of each of the three enhancements to the NRLA model are contained in Chapters IV thru VI. Finally Chapter VII provides conclusions which can be drawn from this study as well as recommendations concerning future applications of the NRLA model.

II. Literature Review

Introduction

The real-world nature of the thesis subject directed the research effort into two principle areas: first from a practical viewpoint, a study and understanding of current USAF policies and procedures regarding Repair Level Analysis (RLA) was necessary and second, from a theoretical perspective the investigation of alternative state-of-the-art optimization techniques which could be used to solve the multi-item multi-echelon repair level decision problem was required. It was essential that each of these subjects be addressed to ensure that products generated from this effort be not only acceptable in terms of their technical content and structure, but also be compatible with existing current USAF logistics policy and doctrine.

Repair Level Analysis - A Macro View

In an effort to assist Air Force acquisition managers in the successful design and implementation of efficient maintenance structures, the Department of the Air Force has established specific guidelines and procedures to be followed by both the government and contractors in the design, analysis, and operation of the RLA program (14:1). As stated in AFSC/AFLC Regulation 800-28, Repair Level Analysis (RLA) Program, " the RLA program has two major objectives:

(1) Design-oriented RLA. This is the preliminary analysis that begins in the conceptual phase of the program and continues through the critical design review. Its goal is to evolve a design that considers the economics of support alternatives and produces an economical life cycle cost profile.

(2) Provisioning-oriented RLA. This analysis begins during the full-scale development (FSD) phase and continues into the production program. The objectives of this phase of RLA are to assign the maintenance portion of the source, maintenance, recoverability (SMR) codes and complete maintenance planning during the provisioning process."(14:1)

Both of these portions of the RLA program are conducted using mathematical models and techniques to determine the most appropriate economic level of repair for system components. However, as can be seen from the causal diagram in Figure 3, there are a multitude of factors which affect the operation of a logistics structure. This is not only reflected in terms of its costs, but also in terms of its requirements for; specialized equipment, manpower with certain maintenance skills, and dedicated facilities at the repair site. Unfortunately, individual identification and assessment of these factors on a system's operational effectiveness is only an initial step in the process of developing a total systems perspective of the repair level problem.

Simulation models can be effective tools to evaluate various integrated logistics management approaches because of their ability to treat complex interactions, time-dependent behavior, and system feedback mechanisms; however, they do not provide 'optimal' solutions but rather 'acceptable' solutions. Conversely, purely analytical approaches, while

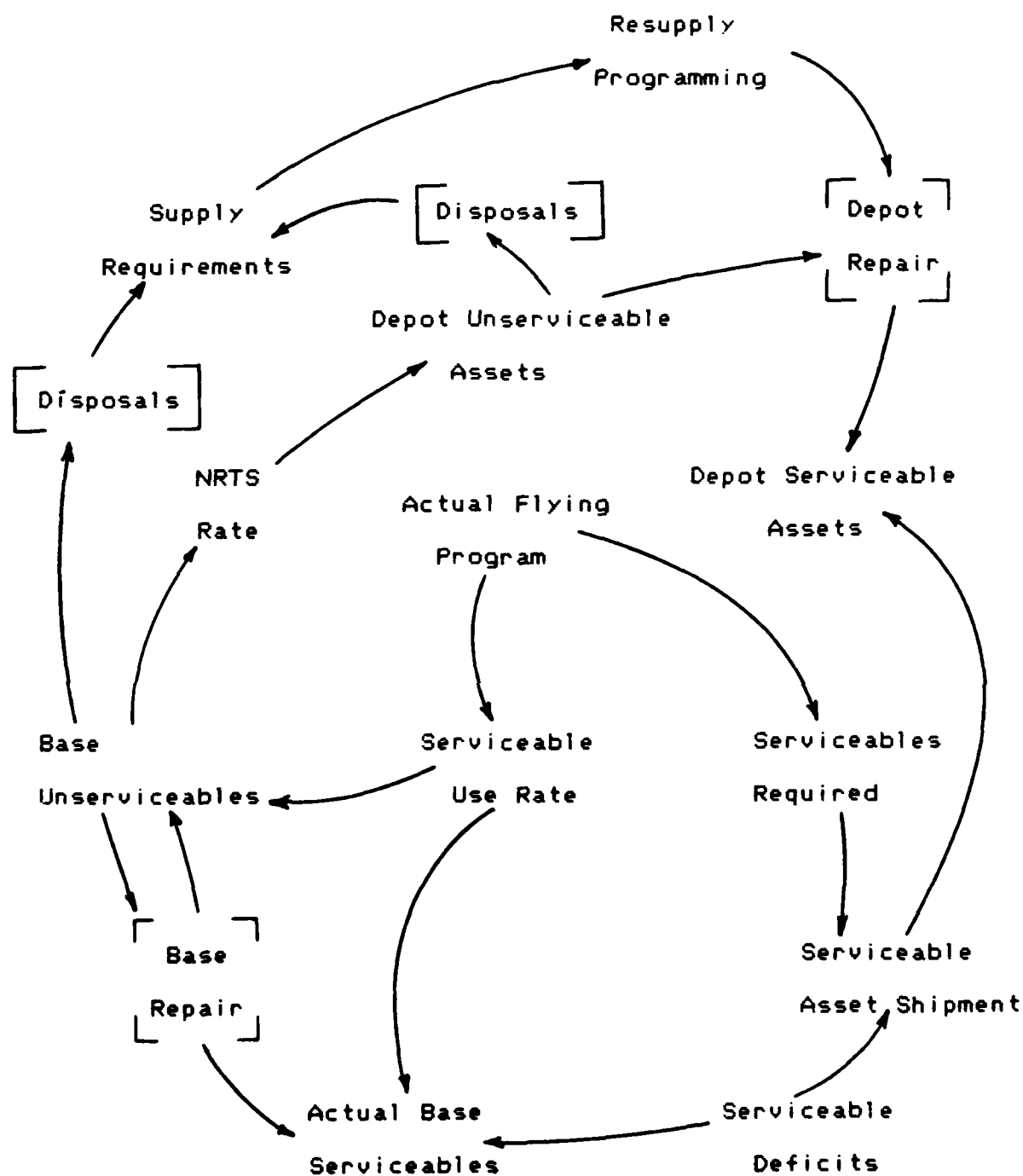


Figure 3. Reparable Assets Structure

proving accurate and appropriate for establishing stockage requirements during the initial provisioning process, are subject to criticism because of their limiting assumptions. The most damaging of these is that of applying steady-state approaches to dynamic systems. Practically speaking, once the system begins to operate structural shifts may occur requiring changes to the repair structure (22:391-392).

Aware of the inherent limitations experienced when applying analytical optimization techniques to dynamic systems such as the multi-item multiechelon repair system, the Air Force has initiated several efforts to assess, on a comparative basis, the life cycle costs (LCC) associated with selection of a repair level for a particular item within a system. Initial efforts were centered on single item repair analysis; however, as has been explained earlier in Chapter I, the inability to adequately address assignment of the fixed costs related to such items as support equipment severely damaged the creditability and validity of this technique.

During the early 1970s, the Air Force began to investigate the feasibility of using network theory as a basis for solving the repair level problem. Much of the motivation for this effort was generated by an article by J.M.W. Rhys' entitled, "A Selection Problem of Shared Fixed Costs and Network Flows" (20:3). Although Rhys does not specifically address the fact that the repair level decision problem could be structured as a special case of the shared fixed cost

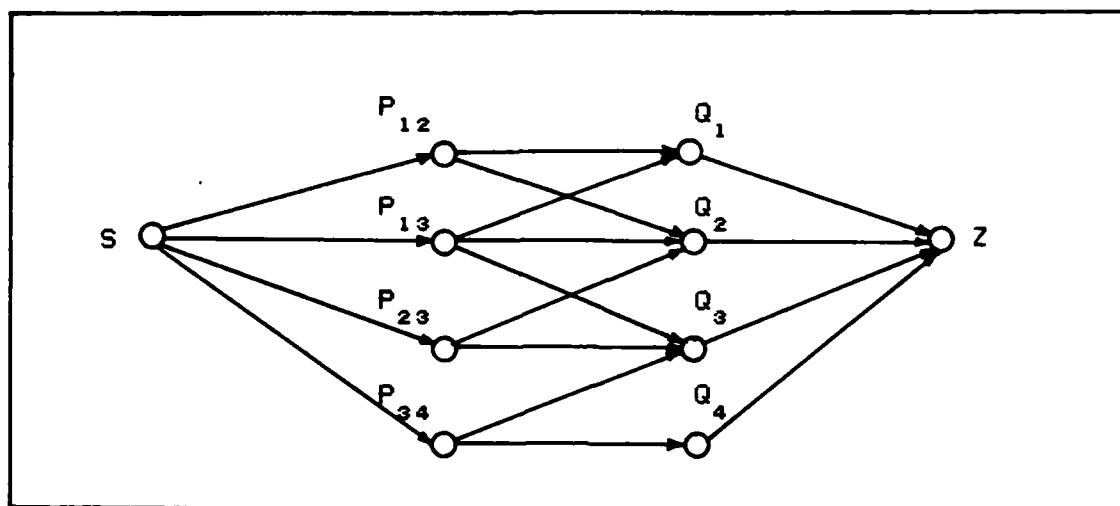


Figure 4. Shared-Fixed Cost Network

problem, he is generally credited with making the major transition from theory to application. The shared fixed cost problem can be described as the problem of selecting a set of activities from some larger set of activities, which require the allocation of fixed overhead costs that cannot be specifically related to one particular activity (21:201).

Using the duality principle of linear programming, Rhys hypothesized and proved that the shared fixed cost problem could be solved by creating a directed network as shown in Figure 4. By determining the maximum flow on this network the minimum cost decisions are simultaneously identifying the minimum cut set. The network has a single source, S and a single sink, Z . Sets of arcs $[SP]$ and $[QZ]$ are defined with each arc in SP corresponding to an activity (benefit), P , and each arc in QZ to a facility (cost), Q . Each arc in SP is allocated a capacity p , where p is the net gain associated with option P , and each arc in QZ is allocated a capacity c ,

with option P, and each arc in QZ is allocated a capacity c, where c is the net loss associated with option Q. (21:201).

The MITRE Corporation of Bedford, Mass. has successfully employed this technique on a variety of Air Force programs to identify optimum levels of repair for weapons systems items (20). Although MITRE's efforts were commendable, as late as 1978, a standardized software package implementing these procedures on an Air Force wide level was still missing. Being the office of primary responsibility (OPR) in the USAF for the development of analytical techniques such as this, the Concepts and Analysis Division, Air Force Acquisition Logistics Center (XRS/AFALC) initiated efforts to design, validate, and distribute just such a product. By June of 1980, a FORTRAN IV computer program known as the Network Repair Level Analysis (NRLA) program had been produced. In conjunction with this program, two documents, The NRLA Users Guide and The NRLA Programmers Guide, were published to assist in the understanding of the structure, limitations, and capabilities of the NRLA program. The Users Guide provides a general description of the model design, the programs operation in terms of input and execution requirements, an explanation of the LRU, SRU, and SE cost computations, as well as, a presentation of the network's formulation. The programmers guide is designed as an aid to understand the " programs structure, logic, input and output operations and the organization of data so that modification and/or corrections can be made " (2:1). One of the most useful portions of the programmers guide is Appendix A: the variables

dictionary, which is an alphabetical listing of the FORTRAN variable names and arrays used throughout the NRLA program.

After reviewing the existing literature relating to the development and application of networking theory to repair level analysis in the Air Force, state-of-the-art solution algorithms were researched. These different techniques were investigated to determine the potential for and the appropriateness of their application within a specialized structure such as the RLA network. Initially the Max-flow Min-cut problem, which is the theoretical basis for the NRLA formulation, was studied. The various solution procedures which appeared to have potential are discussed in the following pages.

Max-flow Min-cut Problem

The max-flow min-cut theorem which was initially established in the late 50's and early 60's by Ford and Fulkerson is the basis for much of the success that has been experienced in formulating an allocation problem as a directed network. Given a network with a source node designated by 's' and a sink node designated by 't', the theorem is as follows: "For any network the maximal flow value from s to t is equal to the minimal cut capacity of all cuts separating s and t" (17:11). The following example should aid in understanding of this theorem. Consider the directed network shown in Figure 5. In this example the maximum flow on the network is 4 units, with 1 unit of flow on path s-1-

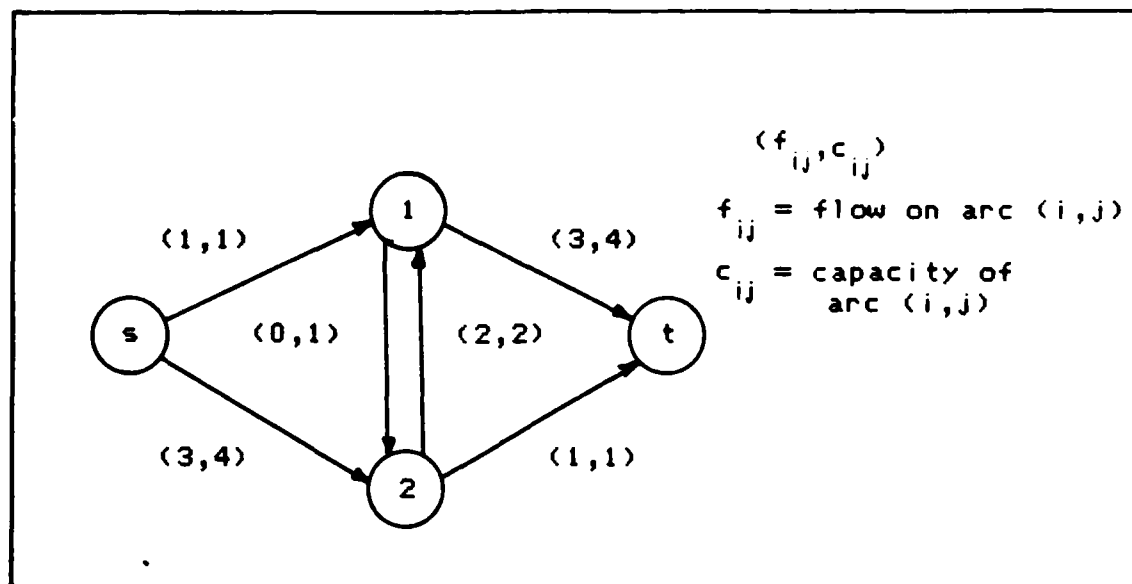


Figure 5. Example Max-flow Min-cut Network

s-2-1-t. The set of arcs $(s,1)$, $(2,1)$, and $(2,t)$ has capacity 4 and is the minimum cut for the network. This set of arcs comprises the critical flow path in the network; to increase flow on the network the capacity of one of these arcs must be increased. Thus the capacity of the minimum cut is equal to the maximum flow.

Mathematical Formulation

The relationship between the max-flow problem and the min-cut problem is founded in the duality relationships of the two problems. The mathematical formulation of these problems will be presented here, for an in depth treatment of the duality relationships refer to one of the following works; Ford and Fulkerson (17:26-30), Bazaraa and Jarvis (5:473-477), or Jenson and Barnes (2:147-153).

Max-flow Problem. Consider a network with n nodes and m arcs, with each arc (i,j) having a lower bound of 0 and an upper bound (arc capacity) of c_{ij} . Let v represent the flow from node 1 (the source) to node n (the sink). Then the maximal flow problem can be stated as follows:

Maximize v

$$\text{Subject to } \sum_{j=1}^n f_{ij} - \sum_{k=1}^n f_{ki} = \begin{cases} f & \text{if } i = 1 \\ 0 & \text{if } i \neq 1 \text{ or } n \\ -f & \text{if } i = n \end{cases}$$

$$f_{ij} \leq c_{ij} \quad i, j = 1, 2, \dots, n$$

$$f_{ij} \geq 0 \quad i, j = 1, 2, \dots, n$$

where the sums and inequalities are taken over all of the arcs in the network.

Before proceeding to the min-cut problem, the concept of a cut-set needs to be explicitly defined. Let N be the set of nodes contained in a network. Let X_1 be any subset of N containing node 1 (the source) but not node n (the sink). Similarly let X_n equal $N - X_1$, the subset of N containing node n but not node 1. Then the set of arcs connecting these two subsets is called the cut-set. For other than trivially small networks, to determine the number of possible cut-sets becomes a problem of a combinatorial nature. The min-cut problem seeks to identify the cut set with the minimum capacity.

Min-cut Problem. The dual to the max-flow problem is

the min-cut problem. Two additional variable sets are added in the dual, π_i which corresponds to the conservation equations and δ_{ij} which can be thought of as "identifying variables". They are identifying variables because in the optimal solution if $\delta_{ij} > 0$ then arc (i,j) becomes a member of the bottleneck set of arcs defining the cut-set. It is not necessary for the cut-set to be a unique set of arcs. The min-cut problem can be defined as follows:

$$\begin{aligned} \text{Minimize} \quad & \sum_{i=1}^n \sum_{j=1}^n c_{ij} \delta_{ij} \\ \text{subject to} \quad & \pi_n - \pi_1 = 1 \\ & \pi_i - \pi_j + \delta_{ij} \geq 0 \quad i, j = 1, 2, \dots, n \\ & \delta_{ij} \geq 0 \quad i, j = 1, 2, \dots, n \end{aligned}$$

Labeling Algorithms

The max-flow problem is usually solved using some type of labeling algorithm. There are a multitude of these algorithms available to solve this problem; however, many of these are the result of simply changing the decision rules for selecting the flow augmenting paths. The classical approach normally used to solve this problem, was developed by Ford and Fulkerson and uses a "Breadth First Search" algorithm as its basis. This approach has been fully developed and applied to the max-flow problem. Additionally, several refinements to this algorithm have been developed and will be discussed. Another approach to this problem bases its solution procedure on a "Depth First Search" algorithm as its basis. The conceptual basis for the depth first approach is

presented in Horowitz and Sahni (19:268-269). However, the literature review did not reveal an application of this search algorithm to solving the max-flow problem; therefore an algorithm to accomplish this was developed as part of the thesis effort. A brief description of these algorithms will be presented with an in-depth description reserved for Chapter V.

Before proceeding to the labeling algorithms, a brief review of common terminology is in order. In both algorithms labels are assigned to the nodes of the network. Initially, all nodes except the source node are unlabeled. When a label is assigned to a node the label contains two pieces of information: the node the flow came from, and the amount of flow potentially available. Consider the label $(1+,5)$, this label indicates that 5 units of flow is available from node 1. The label $(2-,3)$ indicates that 3 units of flow which had previously been flowed to node 2 has the potential to be returned to another node for redirection. It should be noted that the labels only indicate the potential amount of flow which may pass between nodes. The actual amount of flow and the path it will take are not determined until a path to the sink node is found.

The node's label identifies the node as being in one of three states. State 1 is unlabeled, state 2 is labeled but unscanned and state 3 is labeled and scanned. The process of scanning occurs when a node has all of the arcs originating or terminating at it checked for available additional flow.

This is often referred to as checking for admissible arcs. Admissible arcs occur in two forms: forward arcs and reverse arcs. An admissible forward arc originates at the node being scanned, has a destination node which is unlabeled, and has current flow less than capacity. An admissible reverse arc terminates at the node being scanned, has a source node which is unlabeled, and has current flow greater than the lower bound on the arcs capacity.

Breadth First Algorithm. Initially, all nodes are unlabeled (state 1) except the source node which is labeled and unscanned (state 2). The source is then scanned and all admissible arcs are identified. Their associated nodes are labeled and their state changed to state 2. After each node is labeled, a check is made to determine if the sink node has been labeled. If it has, the algorithm goes to a flow augmenting procedure; if not, it continues. When the source has been scanned, a labeled but unscanned node is selected (usually in numerically ascending order) and is scanned. This continues until the sink node is labeled or there are no available nodes in state 2. If there are no nodes in state 2, then the maximal flow has been identified and the algorithm stops. When the sink node is labeled, the flow augmenting algorithm comes into play. It identifies the amount of flow which could be flowed to the sink from its label. It then uses the labels to trace the path back through the network to the source. Each arc on this path has its flow adjusted by the amount which actually reached the

sink node. Once this flow augmentation is complete all labels are erased and the algorithm restarted.

Since arc capacities are restricted to being only integer values, termination of this algorithm will occur in a finite number of iterations. Ford and Fulkerson (17:121) provide an example that shows that the algorithm may not terminate or terminates with a sub-optimal flow when irrational arc capacities are allowed. However, a problem can still arise when using integral capacities. This involves the situation where the total number of flow augmentations is equal to the maximum flow on the network. Edmonds and Karp (16:250) present an example where the flow augmentation at each iteration is only one unit, this leads to an upper bound being created for the number of augmentations equal to the max-flow of the network.

Two refinements of the algorithm are presented by Edmonds and Karp (16:251-253) which can reduce the upper bound on augmentations required. The first refinement involves decision rules for selecting the shortest paths for flow augmentations. They show that if each flow augmentation is done along a path with the fewest number of arcs then a max-flow will be identified in at most $1/4(n^3 - n)$ augmentations.

This result is derived based on the length of the paths to a bottleneck arc from the source and the sink, and the max number of occurrences of bottleneck arcs in a network of n nodes. A further refinement to this process, includes using a rule where the weights of the paths are calculated based on

their lengths and these weightings are then used to aid in the selection of a path. Another rule involves selecting the path with the fewest number of reverse arcs.

A second refinement presented uses a decision rule which selects the path with the maximum possible number of flow augmentations (16:253-255). This refinement provides that the maximum number of augmentations necessary to attain a maximal flow will be at most $1 + \log_{N/(N-1)} f^*(t,s)$, where $f^*(t,s)$ denotes the value of maximal flow. Both of these refinements are intuitively appealing in that the selection of a shortest path or the max-flow per augmentation seem to be good rules of thumb for obtaining max-flow in the fewest number of iterations. Additional considerations for speeding up the process could involve: not erasing the existing set of labels after each iteration, or simply saving the location of the last node with excess potential on the flow augmenting path. This could save a great deal of time which is expended doing redundant labeling and should significantly improve computational times.

Depth First Algorithm. As in the "Breadth First Algorithm" the initial states of the nodes in the "Depth First Algorithm" are source node in state 2 and all other nodes in state 1. The search for a flow augmenting path initiates at the source by identifying an admissible forward arc. The node associated with this arc is labeled and its state is changed to state 2. A search is then made from this node for an admissible arc and the process is repeated until

the sink is labeled or no admissible arcs are available to augmenting program is called and flow is augmented along the path. As the flow is augmented on each arc a check is made to determine if there is any potential flow remaining at any of the nodes on the path. If there is no flow potential at a node, that node's label is erased and it is returned to state 1. When this augmentation is complete, the algorithm then starts with the last labeled node on the path and proceeds to find another path to the sink. If the sink cannot be reached from the last node on the path; and there are no admissible arcs at this node, then the algorithm backs up to the previous node on the path and checks for admissible arcs. This process continues until the sink is labeled or until the source node is revisited, at which point a new admissible arc is chosen from the source and the algorithm starts again. When there are no further admissible arcs at the source and the last path has been retraced back to the source, the maximal flow has been found and the algorithm halts.

To ensure that the maximal flow is obtained and to identify the unique set of labels associated with this flow, it is necessary to make three passes through the network with this algorithm. During the first pass only forward arcs are considered. The result of initially restricting the flow along just these arcs is that the majority of the flow reaches the sink node via the shortest available paths. Additionally, it creates the potential for redirecting flow on a reverse path on the second pass. The labels are all

erased at the end of the first and second passes so that all possible paths can be explored during the subsequent passes. The second pass is used to identify all additional paths through the network on which flow can be augmented. These paths may contain reverse arcs. The third pass is used to identify the unique set of labels associated with the minimum cut set.

Minimum Cut Algorithm. As explained earlier, the optimum set of repair level decisions can be determined by finding the minimum cut set of the RLA network. Because of this relationship, an algorithm which could directly solve the min-cut problem would be acceptable for use with RLA networks. A search of the literature indicated that the most promising algorithm of this type was developed by Dessouky and Phillips (15) and is called the "Cut Search Algorithm". In this algorithm the min-cut set is located directly by a cut seeking procedure.

The min-cut set, which is designated by $K^*(s,t)$, is identified by a two stage process which divides the set of network nodes, designated by N , into three groups T , W , and S . The first stage of the process starts by assigning the sink node to set T . Set T is then expanded toward the source by computing the value of its cut set defined by $K(\bar{T}, T)$ where $N = \bar{T} + T$. At each iteration a calculation is performed for every node which is a member of \bar{T} and has an arc connecting it to T to determine if it can be added to T . This stage terminates when \bar{T} contains only the source node or if no

nodes in \bar{T} can be added to T . If \bar{T} contains only the source node then $K(\bar{T}, T) = K^*(s, t)$, and the minimum cut set has been identified. If T cannot be expanded to include all nodes except the source then the second stage is initiated. In this stage the sets W and S initially contain only the source node. Set W is expanded toward set T until $\bar{W} = T$ at which point the min-cut set is contained in S . S is contained in W ; however, it is expanded only by adding nodes of W , which will reduce the value of the cut-set as defined by (S, \bar{S}) . Therefore at termination $K^*(s, t) = K(S, \bar{S})$.

Computational results comparing this algorithm with the Out-of-Kilter Algorithm (OKA) developed by Fulkerson and Danzig are presented in Dessouky and Phillips' article (15:403). These results indicate that the Cut Search algorithm is more efficient than the OKA. As an example; for a network consisting of 16 nodes and 240 arcs the OKA took 5.33 seconds of cpu time while the cut search algorithm took 1.58 seconds (15:403).

Although the Cut Search Algorithm appears to be very efficient, investigation of its applicability to the RLA problem was not addressed during this research effort. This algorithm was reserved for investigation in the event that the depth first labeling algorithm proved less efficient than anticipated and to allow for the investigation of the applicability of generalized network primal algorithms described in the next section.

Generalized Network Primal Algorithms. In recent years significant advances have been made in the development and implementation of a class of algorithms which can solve large scale capacitated transportation and transshipment problems. These algorithms, often referred to as primal network codes, can also be applied to the solution of the max-flow problem. Because of this characteristic and the potential for applying NRLA to large size RLA systems such as the B1-Bomber and MX-Missile programs, investigation of one of these primal network codes seemed appropriate. The code chosen for use in this context was a commercially available code called GNET. This code was selected for two reasons; (1) it was currently available on the computer systems accessible to AFIT and (2) it was the subject of an excellent article in Management Science by Bradley, Brown and Graves (7) which covered in detail its design, implementation and use. This type of in-depth coverage of a commercially marketed algorithm is usually very difficult to find or is non-existent.

Due to the complexity of the GNET code, and the fact that it was applied as a subroutine in NRLA to solve the max-flow problem only a brief description of the GNET code will be given. For an in-depth explanation of this particular application of primal networking one should refer to the above referenced article. Basically, the GNET code solves the general linear programming (LP) problems (transportation, transshipment, and max-flow) by specializing these LP prob-

lems into a primal network model. The basis for solving this network is the bounded variable revised simplex method. The uniqueness of this approach however, lies in the fact that the network which is developed by the algorithm takes advantage of the sparsity of the arc node matrix and uses upper triangularized basis representations to quickly solve the network.

Results published by Glover and Klingman indicate that when this code is used for finding solutions to transportation and transshipment problems, that it is 30% to 40% faster than the OKA code in the solution of transportation and transshipment problems. Bradley, Brown and Graves indicate that it is currently believed that primal implementations are faster and require less storage than OKA or other algorithms used to solve these type problems (7:3).

III. Methodology

Introduction

Initially, a study was made of the logic and rationale for each part of the NRLA program. The NRLA program, originally developed on the AFLC Honeywell 635 computer using the CREATE time sharing system is "composed of a main routine, a block data subroutine, plus twelve additional subroutines" (2:1). The logic diagram in Figure 6 displays the interrelationships of the various subroutines to the main program. Appendix B provides a brief explanation of their respective functions. After several discussions with both the developers and users of the model (6,9), it was determined that wherever possible a structured programming approach using FORTRAN subroutines would be used to incorporate new or enhanced features into the model. By using this type of approach, there would be minimal impact on current users of the model. Additionally, it was recognized that a strategy of this type would probably facilitate the verification and validation phases of the design process. With this general concept in mind, the identification of the necessary tasks related to accomplishing the thesis objectives was initiated. In conjunction with this effort, a flow diagram was developed to order and prioritize these tasks. This was done to better visualize how they the total process. This conceptual process resulted in the flow diagram shown in Figure 7. To successfully achieve the thesis objectives of:

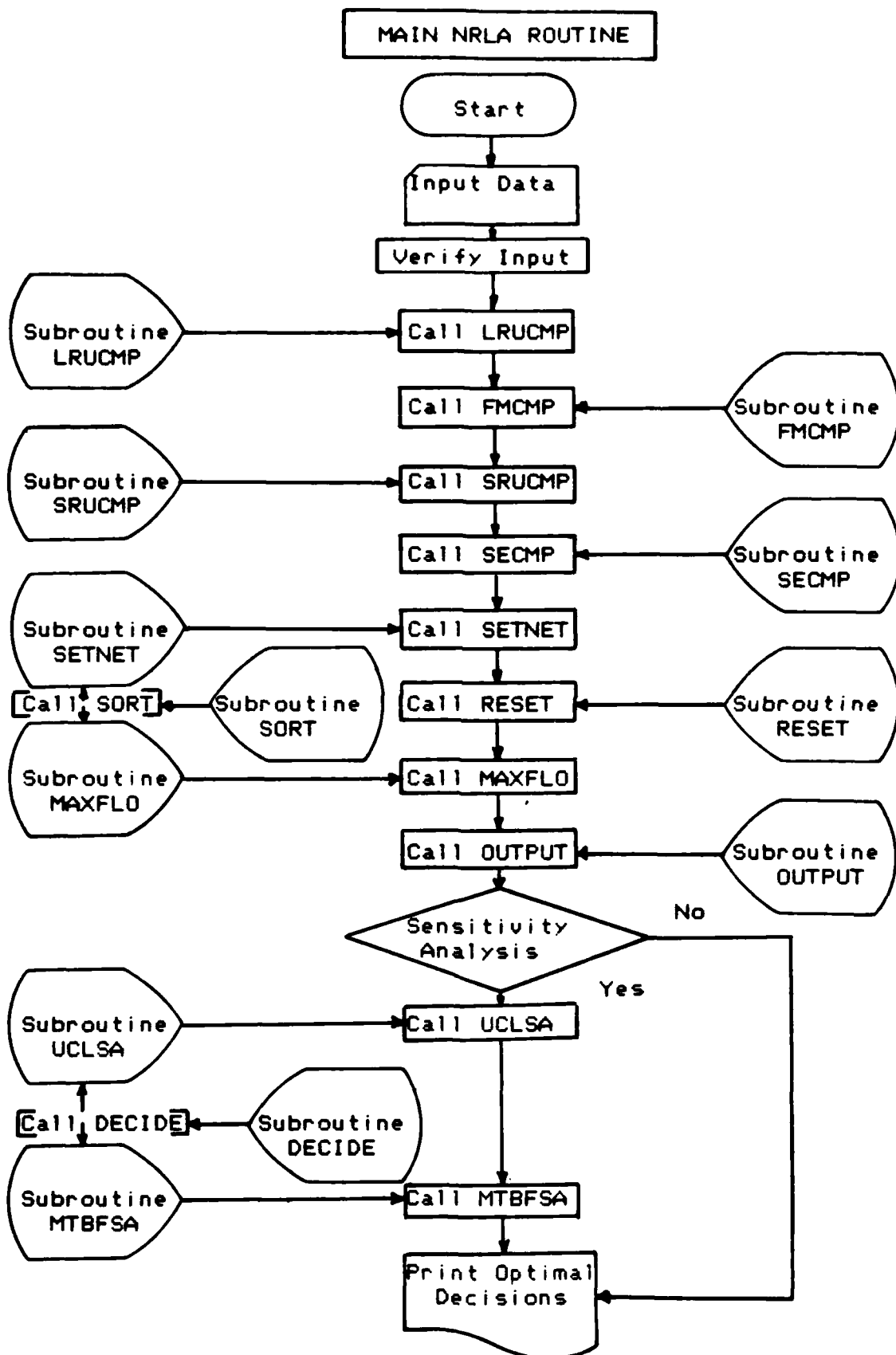


Figure 6. NRLA Program Logic

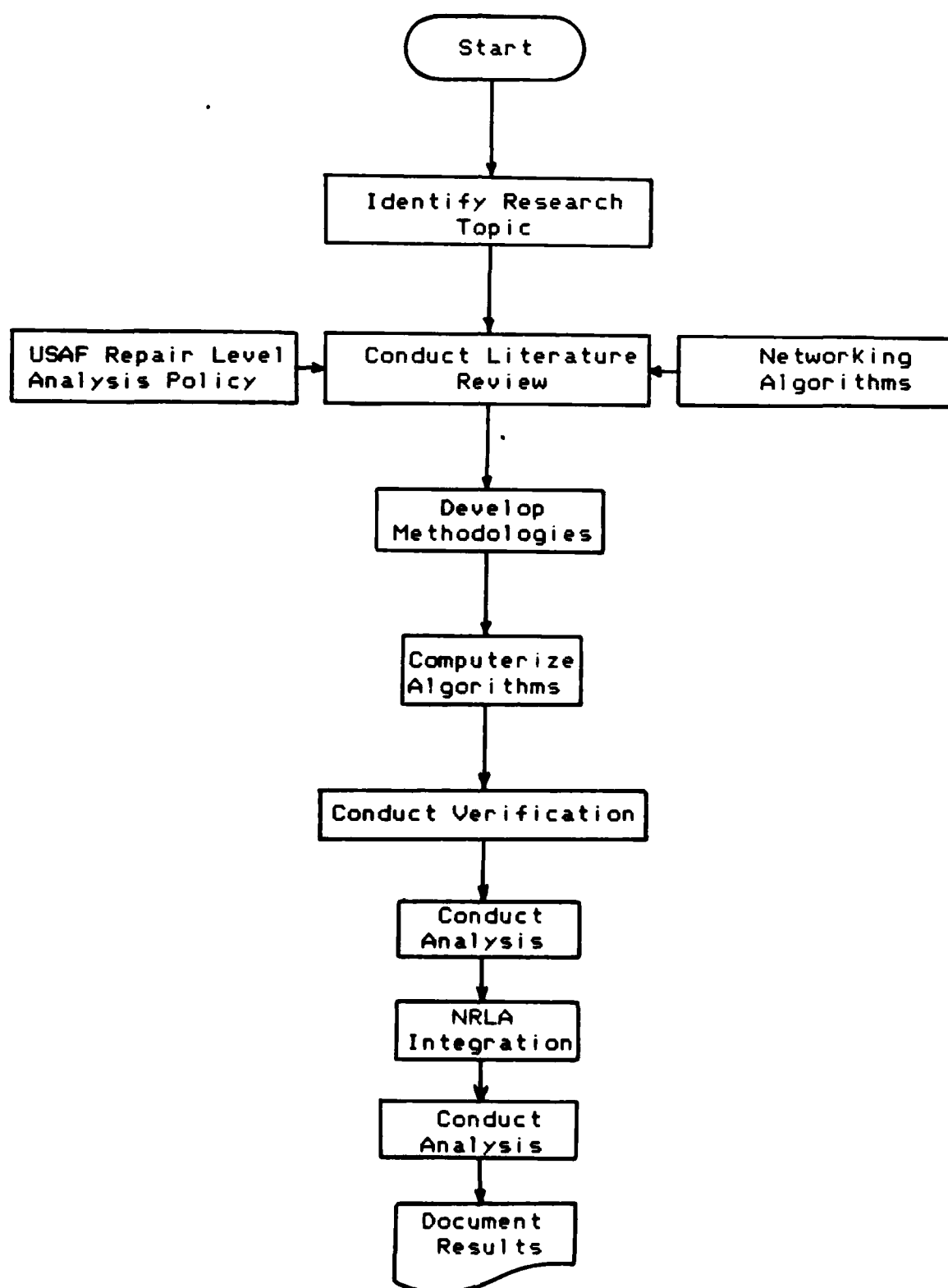


Figure 7. Thesis Task Organization

reducing the NRLA program's execution time, and integrating the CIRF repair level option, four new subroutines were designed and evaluated for possible integration into the existing NRLA structure. These are called MARLA, MSETNT, MAXFLO, and CRFSET and are indicated by the dashed lines in Figure 8.

The MARLA subroutine employs the concept of marginal value analysis in an attempt to determine if optimal decisions can be made for individual items. As can be seen from Figure 8, the MARLA subroutine functions as a preprocessor or filter to the MSETNT subroutine. In effect, when MARLA makes an optimal decision for an LRU or SRU failure mode, it has reduced the total number of items which need to be further analyzed. It was believed that gains in time efficiency would be achievable from this enhancement; the network could be built faster in MSETNT, and the problem to be solved in MAXFLO would be of smaller magnitude. This assumes that the execution time requirements for the MARLA, MAXFLO, and MSETNT subroutines would be less than what is currently required for the SETNET and MAXFLO subroutines in NRLA.

The second area which exhibited a potential opportunity for reducing NRLA's execution time was in the MAXFLO subroutine. MAXFLO's purpose is to determine the minimum cost set of optimal repair level decisions. The solution technique currently employed is a direct application of the two stage labeling procedure developed by Ford & Fulkerson (17:22). Implementation of a new labeling procedure appeared likely to

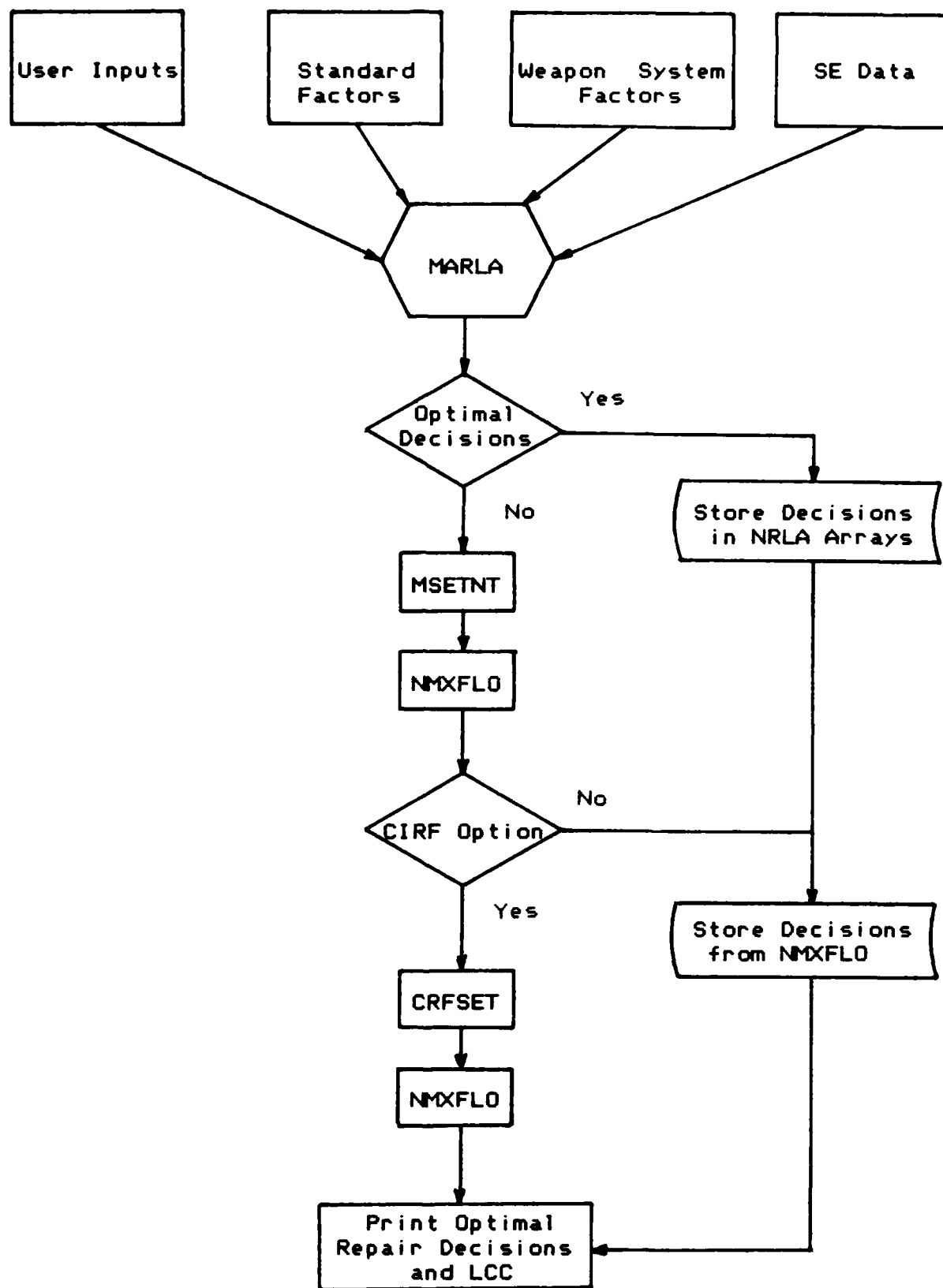


Figure 8. NRLA Thesis Methodology

yield significant time savings in this area of the program. The alternative labeling technique which was developed is constructed in NMXFLO and, as will be explained later, can be used as an alternative to, or in conjunction with, the existing MAXFLO subroutine.

The final objective of incorporating a CIRF maintenance concept as a fourth repair alternative was proposed to be accomplished by using a third subroutine called CRFSET. Subroutine CRFSET was used with the existing SETNET subroutine to determine if the CIRF repair option could be the optimal decision for a particular item. It compares the repair level recommendation generated by SETNET, (depot, base, or scrap) with the CIRF option by building a new unique network (See Figure 8).

Although integration of each of these subroutines would necessitate modifications to other portions of the NRLA program, variable and array integrity was to be maintained to the maximum extent possible. Additionally, the individual subroutines were developed with the objective of being compatible with existing program data structures, as well as, with linkages between data elements (i.e. the use of pointers). Separately, these subroutines accomplish specific subobjectives which contribute to the successful achievement of the previously outlined thesis objectives. The following three chapters will discuss in detail the concept development, computerization, verification, validation, and analysis phases of each of these three major enhancements.

IV. Marginal Analysis Enhancement

Concept Development. The application of marginal cost analysis techniques to solve the repair level decision problem is conceptually based upon the idea that the analyst can use existing cost information that is readily available to quickly identify the optimal level of repair for a specific item. AFSC/AFLC Pamphlet 800-4, Repair Level Analysis Procedures, identifies marginal analysis as a simple and efficient way to synthesize the repair level problem into a more workable format (13:32-33). An appealing feature of using marginal analysis for repair level decision making is its ability to make decisions without prorating the cost of the support equipment which would be necessary to repair an item. This precludes the possibility of making inconsistent or suboptimal decisions. It is for these reasons that development of a marginal analysis approach was selected as a possible solution to solving the problem associated with NRLA's computer run time.

To repair or replace an item or component of an Air Force system requires the expenditure of materiel and personnel resources, as well as the utilization of specialized and sophisticated diagnostic test equipment. The costs which are generated from this process can be broadly categorized into two types of costs: (1) Pipeline costs (P) - which are defined as those costs incurred directly to repair the item, and (2) Support Equipment costs (SE) - which are defined as

Table II
Repair Level Costs

- I. Pipeline Costs
 - A. Initial Spares
 - B. Packing and Shipping
 - C. Replenishment Spares
 - D. Repair Labor
 - E. Item Management
 - F. Technical Management
 - G. Training of Maintenance Personnel
- II. Support Equipment Costs
 - A. Acquisition
 - B. Operation and Maintenance
 - C. Facilities
- III. Scrap Costs
 - A. Initial Spares
 - B. Replenishment Spares

those costs related to SE use that cannot be assigned to a specific item. Table II gives an itemized listing of each of these categories of cost. Once these costs have been determined for each possible level of repair, a marginal analysis can be made of the SE resources at each repair location. In the context of comparing the four repair options of Depot, Base, CIRF, or Scrap, if the scrap cost of the item is known to be less than the pipeline costs of the item at the other locations, then the scrap decision will always be the optimal decision for that item. In this situation, the scrap decision is said to dominate the other three alternatives (13:32). Equations (1) through (3) represent the mathemati-

cal interpretation of the marginal analysis comparisons.

$$\text{If } P_S < P_D \quad \text{then} \quad P_S < P_D + SE_D \quad (1)$$

$$\text{If } P_S < P_C \quad \text{then} \quad P_S < P_C + SE_C \quad (2)$$

$$\text{If } P_S < P_B \quad \text{then} \quad P_S < P_B + SE_B \quad (3)$$

Where:

P_S = pipeline costs to scrap the item.

P_D = pipeline costs to depot repair the item.

P_C = pipeline costs to CIRF repair the item.

P_B = pipeline costs to base repair the item.

SE_D = cost of the depot support equipment.

SE_C = cost of the CIRF support equipment.

SE_B = cost of the base support equipment.

In order to determine the marginal value of the SE resources for each level, the pipeline costs associated with repair of the item are subtracted from the pipeline costs to scrap the item:

$$MV_{SD} = P_S - P_D \quad (4)$$

$$MV_{SC} = P_S - P_C \quad (5)$$

$$MV_{SB} = P_S - P_B \quad (6)$$

From the above computations, it can be seen that in this initial example that the marginal value of the SE at every level was negative. This would mean that there would not be any cost savings to be realized from buying the necessary SE to repair the item. In fact, before the purchase of SE resources could be economically justified, either the

pipeline scrap cost would have to increase or the pipeline costs of one of the repair options would have to decrease until one of the SE marginal values became zero.

Unfortunately, many situations do not lend themselves to such a straight forward type of analysis. A much more comprehensive set of decision criteria must be developed when dealing with SE that possess a positive marginal value. Based on the scenario of having three possible repair locations, as well as, the option to discard the item, a comparison must now be made not only between the pipeline costs of the item; but also between the total costs to repair the item for each of the alternative levels. Table III displays in a systematic fashion the necessary cost comparisons which must be made to ascertain whether or not a repair option can be eliminated. If any of the statements are true for a given repair level, that particular repair option is eliminated. To interpret the meaning of these relationships with respect to the marginal value of the SE, a joint marginal value is calculated as follows:

$$MV_D = \text{Min} [(P_C - P_D), (P_B - P_D), (P_S - P_D)] \quad (7)$$

$$MV_C = \text{Min} [(P_D - P_C), (P_B - P_C), (P_S - P_C)] \quad (8)$$

$$MV_B = \text{Min} [(P_D - P_B), (P_C - P_B), (P_S - P_B)] \quad (9)$$

The joint marginal value is always taken as the minimum of the three values because it identifies the maximum amount of funds that should be allocated for the SE at that repair location. As soon as the cost of the SE exceeds this amount,

TABLE III
Marginal Analysis Decision Criteria

DEPOT

If ($P_S < P_D$) then D'

If ($TC_C < P_D$) then D'

If ($TC_B < P_D$) then D'

CIRF

If ($P_S < P_C$) then C'

If ($TC_D < P_C$) then C'

If ($TC_B < P_C$) then C'

BASE

If ($P_S < P_B$) then B'

If ($TC_D < P_B$) then B'

If ($TC_C < P_B$) then B'

Where : D' = eliminate the depot repair option.

C' = eliminate the CIRF repair option.

B' = eliminate the base repair option.

$TC_D = (P_D + SE_D)$ total repair costs at depot.

$TC_C = (P_C + SE_C)$ total repair costs at CIRF.

$TC_B = (P_B + SE_B)$ total repair costs at base.

it would no longer be cost effective to purchase the support equipment to repair the item (13:71). A deductive type of process is used to isolate the optimal decision. Only if three of the four choices can be eliminated does the marginal analysis approach identify the optimal decision. However, if marginal analysis were to be used as a separate analysis program the utility of its approach should not be based solely on its ability to identify optimal decisions. But rather a more accurate measure would be the total number of repair options it could exclude from the decision making process.

Computerization/Integration. To implement the concept of marginal repair level analysis a FORTRAN IV program called MARLA was developed. Appendix D contains a computerized listing of the MARLA subroutine. This subroutine is designed to be called by the MAIN subroutine subsequent to the computation of the LRU/SRU pipeline repair costs which are performed in the LRUCMP, FMCMP, and SRUCMP subroutines, but prior to the computation of the SE requirements which are made in subroutine SECMP. The reason that the MARLA subroutine should be called in this manner is two fold: first, to accurately perform the marginal analysis MARLA needs the pipeline costs for each of the items, and second, any SE that is purchased as a result of a MARLA optimal decision should be considered when determining the additional SE requirements for the remaining items.

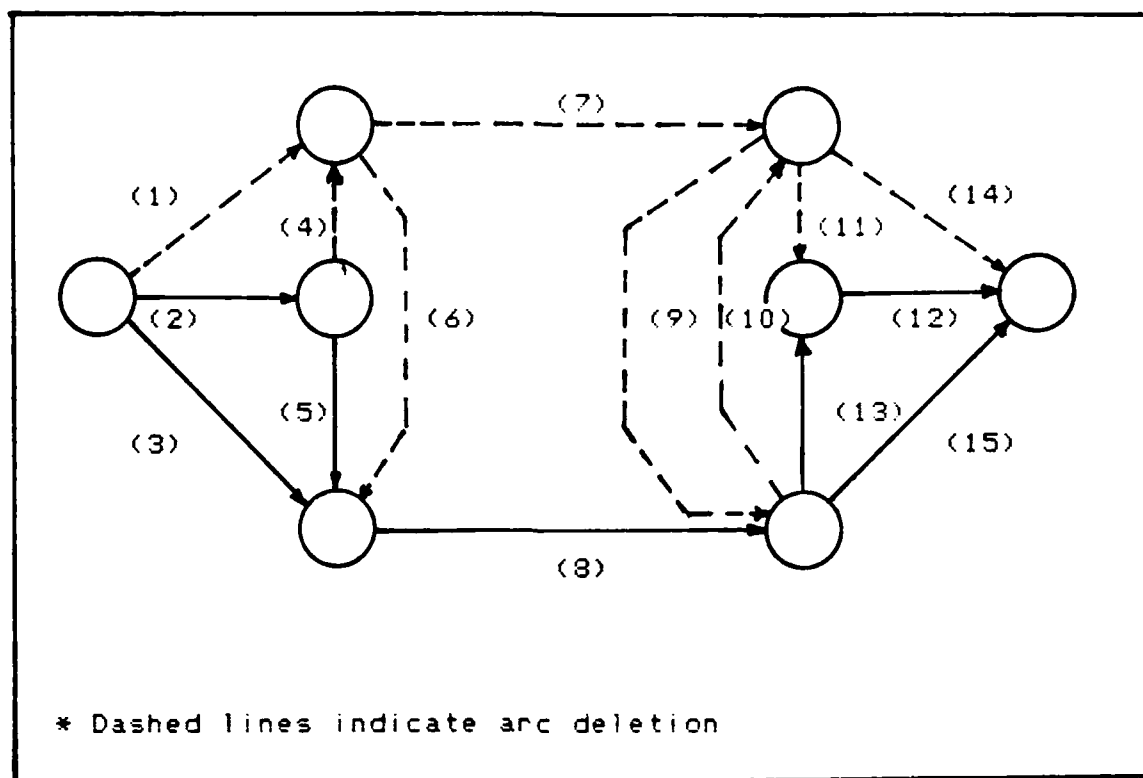


Figure 9. Example of Proposed Arc Elimination Process

The most difficult interface that had to be designed for integration of the MARLA subroutine was the modification of subroutine SETNET to accept the MARLA decisions. When optimal decisions were made by MARLA, it eliminated the requirement to construct the arcs which would be necessary in MAXFLO to analyze the item. The example in Figure 9 indicates the arcs that could be erased from a network if an optimal decision is made; in this case if an optimal decision had been made for LRU #1, eight out of fifteen of the network arcs (#1,#4,#6,#7,#9,#10,#11,#14) are no longer needed. Using this idea, modifications were made to subroutine SETNET to give it the capability to reconfigure a new network based

strictly on just those items still requiring a repair level decision. This new subroutine was called MSETNT. By implementing this change, greater efficiencies were expected to be realized during the programs operation. The code which was developed to accomplish this task is contained in Appendix E.

Because item unique information such as unit cost and SE utilization hour requirements are stored in separate arrays for LRU and SRU failure modes, MARLA was divided into two major sections of code; one to analyze LRU items and another to handle SRU items. Proper control of the NRLA exclusion arrays throughout both of these portions of the program was determined to be extremely important when using MARLA as a preprocessor to NRLA. There were three primary reasons for this. First, if a repair level had been excluded by the user, it is not necessary to perform a marginal analysis for that particular combination of item and repair level. Second, when the actual network is constructed in subroutine SETNET, any repair levels that have been excluded have their respective arc costs set to a very large cost called JUMBO. This prevents them from ever entering the cut set of optimal decisions. However, the MARLA decision criteria were not sufficient to exclude a repair option from further consideration by NRLA. Consequently, the FORTRAN code which was written to accomplish the marginal analysis was required to embody this situation. The third effect MARLA's integration had on the exclusion arrays occurred in subroutine OUTPUT.

Here the exclusion arrays are checked again to see if the user has eliminated any of the repair options. For those items where repair levels have been omitted, the program will print the word 'XCLD'. Two programming changes were required in subroutine OUTPUT to ensure that this operation was performed for only user exclusions and not for exclusions generated by MARLA.

Another important aspect of subroutine MARLA is the identification and acquisition of the correct quantities of SE resources necessary to repair the item. Although when performing the marginal analysis the potential SE costs are calculated for each item, the actual procurement of these resources should not be made unless an optimal decision has been made for that item. For this reason the logic of the MARLA subroutine does not adjust the SE arrays for cost and hours until the optimal decision arrays, OPDECL and OPDECS, are screened to determine if in fact a valid requirement does exist for that particular piece of support equipment. MARLA is structured to buy the minimum number of types of SE needed for each repair level based on the total system requirement. This was accomplished by using the logic described in Figure 10. The pivotal factor in this procedure was the accurate determination of the SE hours currently available for each type of SE at each repair level. This precluded the purchase of more SE than was actually needed. The array, HRAVSE, was created and integrated into the common MARLA block statement to store the available hours for each piece of SE and to

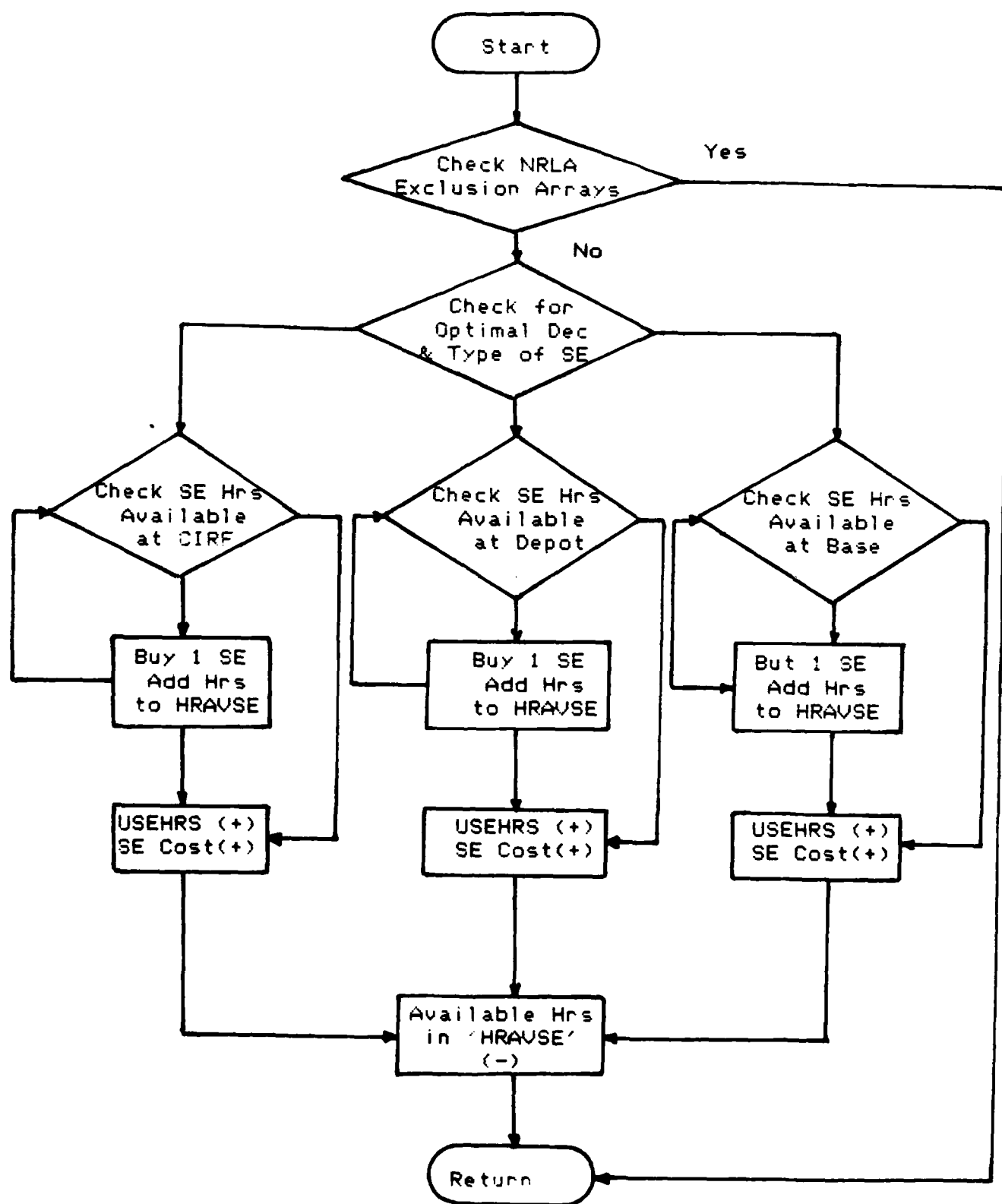


Figure 10. MARLA SE Requirements

allow for these hours to be transferred to the subroutine, SECOMP. This ensured that any SE hours still available could be applied against the utilization hours required for the remaining items to be analyzed.

Verification/Validation. To verify the operational accuracy of the MARLA enhancement, two undertakings were necessary. Not only was it essential to confirm the logic of the MARLA subroutine, but it was also critical to separately authenticate the major changes that had to be made to the SETNET subroutine. It was decided that these tests should be conducted independently to provide the maximum assurance that both portions of the program were performing correctly.

Initial testing of the MARLA subroutine revealed significant program logic flaws. By modifying the subroutine so that it could be run as a separate program errors were quickly identified and corrected. This would not have been possible if MARLA had been immediately integrated into the NRLA program. The sample data set which was used for this testing included a total of 18 LRU/SRU failure modes with three different types of SE at each of the repair locations. Prior to running the program, the correct repair decisions were manually calculated using the decision criteria in Table III on page 46. The results from the MARLA program were then compared to these recommendations. This iterative process continued until the MARLA decisions matched identically with the true decisions. The results obtained from this test indicated that the MARLA subroutine was functionally

correctly both computationally and logically.

A different design was developed to verify the accuracy of the SETNET modifications. A predefined network was utilized to determine if MSETNT would eliminate the appropriate arcs for a given LRU/SRU optimal decision. By presetting the OPDECL and OPDECS arrays with a particular optimal decision, the arcs to be eliminated from the network could easily be identified. A trace of the network, consisting of each arc's source node and destination node, was output to determine if the correct arcs had been removed. Initially, the program eliminated only a portion of the arcs it should have. However, by continuing to use the trace from the network, all of the logic and syntax coding errors were satisfactorily resolved.

During this process, an important point was discovered. If the optimal decision for an LRU is 'base', the arcs representing the transportation costs for replacement SRUs (#6, #9, #10 in Figure 8, pg.48) could also be eliminated. However, since the costs associated with these arcs are related to the repair of the SRU, it was realized that they should be added to the SRU depot and scrap repair arcs (#3 and #8, Figure 8) so that the SRU costs would not be underestimated.

Initial validation of both the MARLA and MSETNT sub-routines was accomplished by consulting with the original developers of the NRLA program (6). However, complete validation of the MARLA enhancement will only be determined after

NRLA users throughout the Air Force and defense contracting community have had the opportunity to exercise this option and provide feedback as to its performance and accuracy.

Analysis/Results. The feasibility of using the MARLA subroutine as a preprocessor to the NRLA program was analyzed in terms of its effect upon total processing time and additional storage requirements. A data set which had previously been analyzed by the original NRLA program was used with the enhanced program. In this way, a standard of comparison could be established to measure the effect on program time and storage requirements. Additionally, the repair level recommendations produced could be compared for consistency.

When used as external preprocessor, the MARLA program produced results identical to the NRLA program. However, when integrated as a subroutine in the main program, the repair recommendations generated did not agree. It was observed that although the pipeline logistical costs for each repair level matched exactly for both programs, the SE cost estimates differed significantly. It was determined by cross checking the outputs from both programs that this was probably the reason for the disparity in the model's recommendations.

Two other significant results were observed from comparing the output of the two programs. These related to the processing time and the number of arcs constructed in each of the networks. These program parameters are directly proportional to each other; the greater the number of arcs in the

network, the longer it take the program to execute. For the sample data set, which consisted of 23 LRU failure modes, 10 SRUs, and 12 pieces of SE, the original NRLA network consisted of 361 arcs and required .0149 hours to execute. For the enhanced program the output indicated that the network contained only 12 arcs and processed in .0034 hours. Although these results showed a substantial improvement over the original network, a high level of confidence could not be placed in their validity because of the programs inability to produce decisions consistent with the original NRLA program.

Storage requirements for the new program were also analyzed. To incorporate the MARLA subroutine nine new arrays were required. Seven of these were needed to perform the marginal analysis while the two remaining ones, HRAVSE and TSECST, were used to collect the SE cost and available hours, respectively. The number of arrays was a function of the number of repair locations. The size of the arrays used in the marginal analysis was determined by the total number of failure modes and SRUs to be analyzed. Appendix A contains a listing of each of these arrays and their function in the program.

Based on the limited test results which were achieved with MARLA during this study, further efforts are necessary in order to fully assess its capability as a preprocessor to the NRLA networking process. A better evaluation could then be made of its utility in terms of reducing the overall program processing time versus the additional storage requirements to achieve this.

V. Max-flow Min-cut Labeling Enhancement

NRLA Network Formulation

Network Structure. Understanding the operation of the NRLA program requires an in-depth knowledge of its network structure and how this structure models the cost factors associated with the repair level decisions. The ten decision cost factors and the eleven logistics cost factors which comprise them are shown in Table I on page 8. As can be seen in this table, the decision cost factors are simply the summation of the logistic costs associated with a LRU's or SRU's repair level decision. This consolidation of costs allows their use as capacities for the arcs of the repair level network. The network formulation was examined briefly in Chapter I; however, the details of how the network is built and how the various cut-sets (repair level decisions) are determined were not covered at that time. The following section covers these topics in detail.

The NRLA program builds the network structure by first assigning numbers to the nodes in a specific order and then laying in the arcs associated with these nodes. As shown in Figure 1 on page 7, the nodes are assigned numbers in the following order source node, depot support equipment, depot LRU failure modes, depot SRUs, base LRU failure modes, base SRUs, base support equipment, and finally the sink node. This careful ordering of the nodes allows the program to develop forward and backward pointers to the vector arrays containing the LRU, SRU, and SE relationships.

Next the arcs are added to the network. This is also done in an ordered manner starting with the source node and working through the nodes one at a time. For simplicity the LRU failure mode nodes will be referred to as LRU nodes during this discussion. The arcs from the source to the depot SE (depot SE cost) are added first, followed by the arcs to the depot LRUs (depot LRU repair costs) and the SRUs (depot SRU repair costs). The next arcs added are the depot SE to LRU and SRU JUMBO (large capacity) arcs. These arcs shown as the thicker arcs in the figure represent the relationships between the individual depot SE and the set of LRU/SRUs which it repairs. Because these arcs have JUMBO capacity they cannot be capacitated (flow equals capacity); therefore, they allow the total set of LRU/SRUs to share the burden of the SE cost.

The arcs originating at the depot LRUs are added next. First the depot LRU to depot SRU (DEC1) arc is added. Then the depot LRU to base LRU arc (scrap cost). The DEC1 arcs represent a portion of the additional costs associated with the decisions to base repair the LRU and depot repair the SRU. The rest of these costs are contained on the DEC2 arcs. The costs are split to allow for representation of the two way trip associated with SRU depot repair verses the one way trip if the SRU is scrapped when the LRU is base repaired.

The SRU scrap cost arcs are added next. They originate at the depot SRU and terminate at the base SRU. Moving on to the base LRU nodes the following arcs are added. First an

arc from the base LRU to its associated base SRU (DEC2 costs) and then an arc from the base LRU to the sink node (base LRU repair costs). Following these the arcs emanating from the base SRUs are added. The first arc is a JUMBO arc from the base SRU to its associated base LRU. This arc ensures that the decisions to base repair the SRU and scrap or depot repair the LRU cannot be made. Next an arc from the base SRU to the sink node is added (SRU base repair costs).

The last nodes dealt with are the base SE nodes. They have several arcs associated with them. The first arcs added are the JUMBO arcs from the set of base LRUs to the SE which are used in their repair. Next the JUMBO arcs from the set of base SRUs to the SE are added, and finally the base SE cost arc is added from the SE to the sink. Each base SE is considered separately and all the above arcs added prior to going to the next SE. The JUMBO arcs perform the same function for the base SE as they did for the depot SE, that of inter-relating the LRU/SRU sets to support the SE costs.

Minimum Cuts. Once the network is built, the set of repair level decisions are determined by applying a max-flow min-cut algorithm to identify the min-cut set. Figure 11 displays the seven potential RLA network cuts. Table IV identifies the decisions associated with these cuts and the costs incurred for each decision. Recall that a cut divides the network into two sets of nodes: one containing the source node and one containing the sink. The arcs of the cut-set are the chokepoints of the network flow. Cut-set 1 in Figure 11

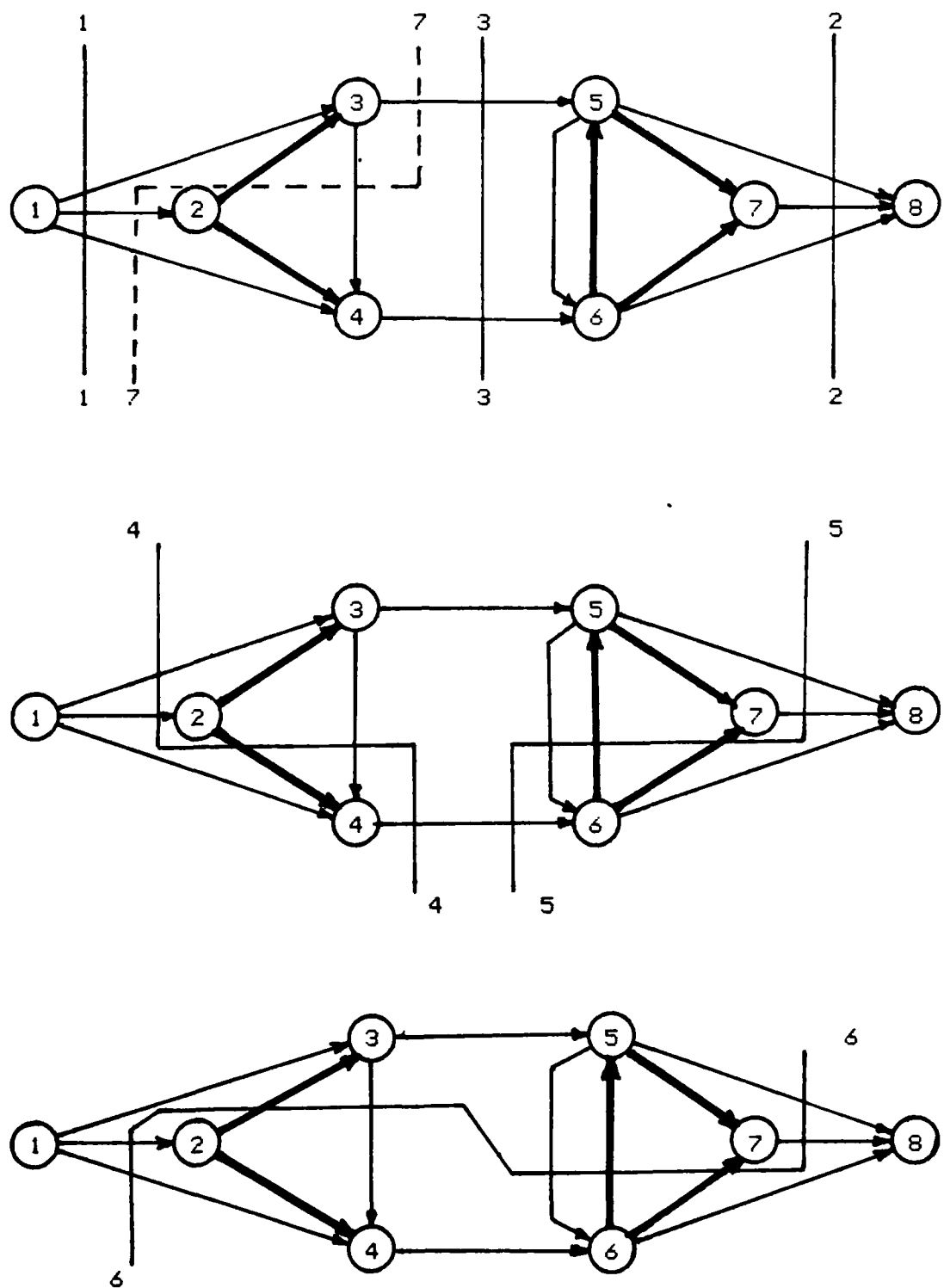


Figure 11. Potential Cuts of the RLA Network

TABLE IV
RLA Decision Costs

RLA DECISION COSTS						
Decision			Included Costs			
CUT #			SE		DEC 1	DEC 2
	LRU	SRU	DEP	BASE		
1	Depot	Depot	Yes	No	No	No
2	Base	Base	No	Yes	No	No
3	Scrap	Scrap	No	No	No	No
4	Depot	Scrap	Yes	No	No	No
5	Base	Scrap	No	Yes	No	Yes
6	Base	Depot	Yes	Yes	Yes	Yes
7	Scrap	Depot	Yes	No	Yes	No

can be seen to contain the following arcs: (1,2), (1,3), and (1,4). These represent the costs associated with the depot SE, the depot LRU repair costs, and the depot SRU repair costs. This indicates that when the maximum flow was determined in the network, these arcs were capacitated; therefore, the optimal repair decisions are depot repair of both the LRU and SRU.

The following notes on the indicated cuts are necessary to ensure that the cut-sets are not misinterpreted. JUMBO arcs cannot be capacitated; therefore, all cuts crossing JUMBO arcs do not include these arcs. Cut-set 4 does not include the DEC1 arc because it originates at an unlabeled node and terminates at a labeled one; to be part of the cut-

set the reverse must be true. Cut 7 represents the anomolous situation where the LRU is scrapped but the SRU is depot repaired. The cost of the SRU is included in the LRU cost, therefore a LRU scrap decision should include scrapping the SRU. The structure of the network cannot exclude the possibility of this cut occuring. If it does occur, a valid decision can be reached by running the model twice: the first time excluding the LRU scrap option, and the second time forcing the LRU and SRU scrap decisions. The optimal repair level decisions are associated with the minimum of these costs.

Now that the RLA network is built and the potential cut-sets explained, it is necessary to examine the efficiency of the algorithm used to solve the network max-flow min-cut problem. The next section covers the labeling algorithms applied to this problem.

Max-flow Min-cut Algorithmic Improvements

Conceptual Development. A primary concern expressed by the users of the NRLA model is the amount of computer resources used if the sensitivity analysis option is selected (6,9). The NRLA program allows the user to perform sensitivity analysis on a selected range of costs and/or failure rates, based on mean time between failures (MTBF), for each LRU, LRU failure mode, and SRU. This allows the user to accomplish a "what if" type analysis to determine the best repair level options in the event that one of these variables changes during the systems development. To

accomplish this sensitivity analysis, the NRLA program modifies each item's cost or MTBF and then re-solves the network based on this modification. It then resets the network and performs the next modification. To get the true impact of this, consider a sample system which consists of 23 LRU failure modes and 10 SRUs with both cost and MTBF sensitivity analysis selected. In this example the sensitivity analysis program would modify and re-solve the network over 300 times. Each time the network is modified, subroutine MAXFLO is called to re-solve it. Therefore, the computational efficiency of the algorithm used in MAXFLO to solve the max-flow min-cut problem plays a major role in determining the total computer run times for the model.

Based on this information, the next step in the research effort involved analyzing the current labeling algorithm for potential time saving modifications. To assist in this analysis a detailed description of the Breadth First Labeling algorithm is presented next (17).

Breadth First Labeling Algorithm

Notation : f_{ij} = flow from node i to node j

c_{ij} = capacity from node i to node j

Concept : Assign labels to the nodes of the network such that a flow path may be determined on the network. Augment flow on these paths until the maximum flow has been attained. The labels have the form $(i+,e)$ or $(i-,e)$, where i is a node of the network and e is a positive

integer or ∞ according to the rules specified in procedure A. The state of a node is modified by the algorithm according to whether it is unlabeled (state 1), labeled and unscanned (state 2), or labeled and scanned (state 3).

Procedure A : Start by assigning the source, s , a label of $(-\infty)$, this indicates that it is receiving infinite flow from an external source. All other nodes are unlabeled (state 1) and the state of the source node is state 2. In general, select any node i which is in state 2. Suppose it is labeled $(k, e(i))$. Scan node i for admissible arcs. Recall that admissible forward arcs require the destination node j to be unlabeled and $f_{ij} < c_{ij}$, while admissible reverse arcs require the source node j to be unlabeled and $f_{ji} > 0$. For all admissible forward arcs assign the appropriate node j the label $(i, e(j))$ where $e(j) = \min(e(i), c_{ij} - f_{ij})$. For all admissible reverse arcs assign the appropriate node j the label $(i, e(j))$, where $e(j) = \min(e(i), f_{ji})$. As each node is labeled a check is made to determine if the sink has been labeled. If it has proceed to Procedure B, if not continue the labeling process. If node i is completely scanned and the sink is unlabeled change node i 's state to state 3 (labeled and scanned). Repeat the general step until either the sink node t is labeled, or until no additional labels can be assigned and the sink is unlabeled. If the sink is labeled go to

Procedure B, if not, stop - the maximum flow has been found.

Procedure B : The sink node t has been labeled $(j+, e(t))$. Trace the labeled path back to the source node s while augmenting the flow on the arcs of this path. In general, if node j is labeled $(i+, e(x))$, replace f_{ij} with $e(t) + f_{ij}$, and if j is labeled $(i-, e(x))$, replace f_{ji} with $f_{ji} - e(t)$. Proceed to node i and continue this flow augmentation until the source is reached, erase all the current labels and return to Procedure A.

Analysis of the above labeling algorithm indicates that it uses the labels it produces very inefficiently. The following simple example will more clearly illustrate this problem. Consider the network shown in Figure 12. The source node is connected to 100 nodes which are in turn connected to the sink. At each iteration of the above labeling algorithm all 100 arcs from the source to the intermediate nodes will be checked for admissibility and the nodes labeled if appropriate. After the source is completely scanned the algorithm proceed to the next labeled but unscanned node and then scans it. In this example one of two things can happen when an intermediate node is scanned; either the sink is labeled or the node is marked as scanned and a new, labeled but unscanned, node is sought. When the sink is labeled, flow is augmented on the path from the source to the sink, all the labels are erased, and the process is restarted. If the algorithm has to scan n inter-

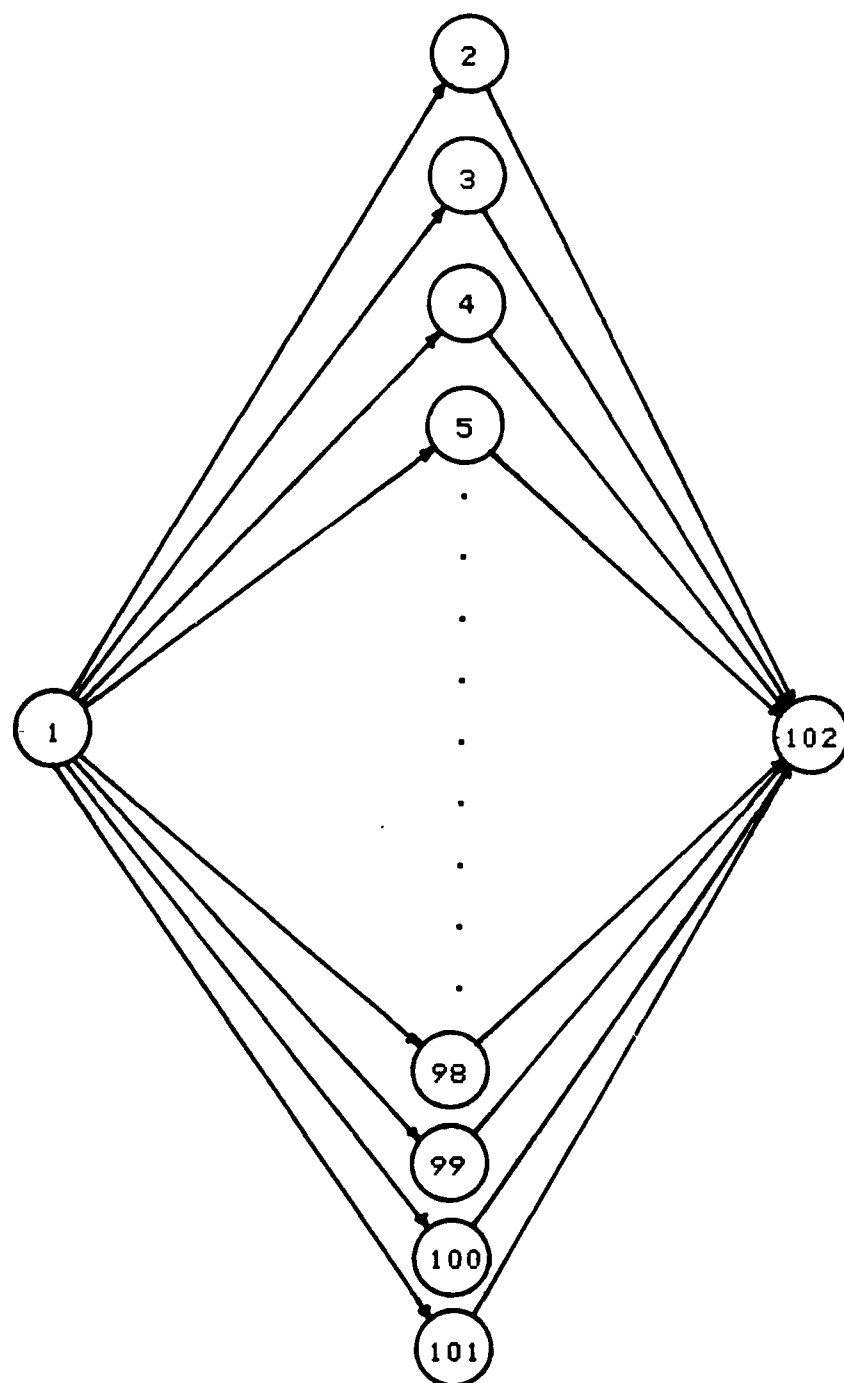


Figure 12. Network Labeling Example

mediate nodes before it finds a path to the sink, then there can be as many as $100-n$ nodes which are labeled and then erased without being used. Now suppose that all arcs of the network initially have flow less than capacity. Given this initial condition, for this example the algorithm would produce 99 excess labels on the first iteration, 98 on the second and so forth, for a total of 4950 excess labels. Clearly this is a less than optimal use of the labels produced.

One way to eliminate this problem involves using a Depth first search algorithm in the labeling process. A review of the literature indicated that the application of depth first search concepts to the labeling process has not been attempted before. Conceptually what a depth first search does, is move through the network from the source to the first level and then directly on to the second level and so forth until the sink is reached. A level in this context consists of the set of nodes whose shortest path from the source is the same length, where length is given in total number of arcs on the path. In the above example this would be equivalent to labeling node 2 and then immediately labeling the sink from node 2, thus eliminating the labeling of nodes 3 through 101. RLA networks are well suited to this type search, in that only two levels separate the source and sink in most cases. The above stated advantages of a depth first search concept led to its use in developing a new labeling algorithm to solve the max-flow problem. A detailed description of this algorithm follows.

Depth First Labeling Algorithm

Notation : f_{ij} = flow from node i to node j

c_{ij} = capacity from node i to node j

Concept : Assign labels to the nodes of the network such that a flow path may be determined on the network. Augment flow on these paths until the maximum flow has been attained. Labels are assigned to the nodes on the path from the source to the sink only. Nodes are scanned only until an admissible arc is found, at which point that node is labeled and scanned for an admissible arc. The state of a node is modified according to whether it is unlabeled state 1 or labeled state 2. Nodes are not labeled as scanned or unscanned.

Procedure A : Initially all nodes are unlabeled (state 1). Proceed by labeling the source node s , $(-, \infty)$, and setting it to state 2.

First Pass (forward Paths Only)

Starting at the source check for an admissible forward arc and label its associated node i $(s+, c_{si} - f_{si})$, assign node i to state 2 and scan it for an admissible forward arc. In general proceed in this fashion until the sink is labeled or the last node on the path has no admissible forward arcs. If the sink is labeled go to Procedure B. If there are no admissible forward arcs back up to the previous node on the path and scan it for an admissible forward arc, if there is one proceed with the general rule, if not continue backing up. When no

admissible forward arcs are available at the source
erase all labels and go to the second pass.

Second Pass (Forward and Reverse Arcs allowed)

Once again label the source $(-, \infty)$ and set it to state 2. Scan the source for an admissible arc and label its associated node. Scan this node for an admissible arc and label its associated node according to whether it is a forward or reverse admissible arc. In general continue in this fashion until the sink is reached or no admissible arcs are available at the last node on the path. If the sink is labeled go to Procedure B. If no admissible arc is available back up to the previous node on the path and proceed. When no admissible arcs are available at the source erase all labels and proceed to the Third Pass. At this point the maximum flow has been identified.

Third Pass (labeling pass)

Repeat the procedure used for the second pass until no admissible arcs are available at the source. When this occurs stop - the max-flow with correct labels to identify the min-cut has been found.

Procedure B : The sink, node t , has been labeled $(j+, e(t))$.

Trace the labeled path back to the source, node s , while augmenting the flow on the arcs of this path. In general, if node j is labeled $(i+, e(x))$, replace f_{ij} with $e(t) + f_{ij}$, and if j is labeled $(i-, e(x))$, replace f_{ji} with $f_{ji} - e(t)$. As the flow is augmented on this

path, each node is checked to see if it has potential flow remaining. If $e(j) > e(t)$ and no previous node on the path has had potential remaining, then this node is stored as the new starting point when the flow augmentation is complete. If $e(j) \leq e(t)$ then change the nodes state to 1 and erase its label. Nodes with potential remaining, stay in state 2 and retain their labels. When the flow augmentation is complete return to the appropriate pass in procedure A and continue the pass by scanning either the stored node or the source.

Several factors contribute to the overall efficiency of the above labeling algorithm. The algorithm pushes the flow to the sink on the first available path, thereby eliminating any unused labels. It retains the position of the deepest node on the with flow potential remaining. This allows for a head start when starting the search for the next flow augmentation path. It also stores an array of the arcs on the path from the source to the sink. This eliminates the need to identify these arcs by individually scanning each node on the path. The breadth first search must identify the path in this way because the entire path is not identified until the sink is labeled.

Computerization/Integration. The coding of the depth first labeling algorithm was accomplished using FORTRAN IV. This version of FORTRAN was used to ensure compatibility with the existing program and to allow for transportability

between computer systems. The fully developed subroutine is called NMXFLO and is listed in Appendix F. The only part missing from this listing are the Common statements from the NRLA program. NMXFLO was designed to replace MAXFLO, and as such can be used in place of or in addition to MAXFLO in the existing NRLA model. The addition of one array (NAPATH) and two variables were the only additions to NRLAs storage requirements.

Verification/Validation. The verification of this subroutine was accomplished by developing a network data set with known arc capacities and flows. This data set was then input to NMXFLO and the resulting arc flows compared with the expected results. In this manner initial logic flaws in the algorithm were identified and corrected. When the algorithm was fully debugged and operating as anticipated it was applied to a validation phase of testing by incorporating it in the NRLA model and using it to solve the max-flow min-cut problem. The results of this testing were compared with solutions using MAXFLO (the original subroutine) for validity. In all instances NMXFLO produced identical results.

Analysis/Results. The final analysis performed on the NMXFLO subroutine dealt with determining its efficiency at solving the max-flow min-cut problem. To accomplish this, the NRLA model was run using data sets supplied by the B-1 SPO and AFALD/XRS. The execution times for the NMXFLO and MAXFLO subroutines were calculated using the CREATE computer

TABLE V
Max-flow Min-cut Algorithm Test Results

Data Set	# LRU	# SRU	# SE	# NODES	# ARCS	Processing Time in seconds		
						(old) Breadth First	(new) Depth First	GNET
without sensitivity analysis								
1	23	10	12	100	361	0.432	0.038	3.041
2	52	0	106	212	470	8.454	0.082	6.054
with sensitivity analysis								
1	23	10	12	100	361	37.764	7.977	**
2	52	0	106	212	470	495.939*	15.419	**
<p>* run time exhausted</p> <p>** GNET was not compared because of the way it was implemented in the model.</p>								

system processing time program. This program clocks the actual processing time in hours. The time hacks were taken at the beginning of the subroutines and just before the return to the main program. The resulting times were then multiplied by 3600 to convert the times to computer processing seconds. The results of these tests are shown in Table V. As can be seen, the depth first search algorithm is

significantly faster than the the breadth first search algorithm. The largest system tested showed the depth first algorithm with 100 fold increase in efficiency over the breadth first algorithm. The results obtained when GNET was used to solve the max-flow problem in NRLA are also contained in Table V. In this case, GNET did not overcome the efficiency of the breadth first algorithm until the system size increased to the maximum size tested, and it never approached the efficiency exhibited by the depth first algorithm. GNET was not tested for the sensitivity option due to the manner in which is was implemented in the NRLA program. The modular form in which GNET was implemented did not allow it to retain information on the network structure. This would result in it taking the total network solution time at each step of the sensitivity analysis. The other algorithms took advantage of previous information and, therefore, any comparison between them in this area would be inaccurate. Based on these results it is strongly recommended that the depth first search algorithm be incorporated as the primary MAXFLO subroutine in the current NRLA model.

VI. Centralized Intermediate Repair Facility Enhancement

Concept Development

To address the feasibility of including a CIRF maintenance option in the NRLA program required an in-depth study of the existing network formulation. The rigid structure of the existing network was not adaptive to the insertion of a fourth repair option. However, it was hypothesized that by constructing a second network, which would represent the optimal decision from the first network and the CIRF alternative, that a comparative economic analysis could be made.

To successfully implement this concept into the program, five major tasks had to be accomplished. First, a definitive description of the meaning of CIRF repair was required. Without this, it would be very difficult to validate the computerization of the CIRF repair option. The second major requirement was the identification of the necessary input data elements to support a CIRF developed network. The third objective related to the development of CIRF cost equations to calculate pipeline as well as support equipment costs. Fourthly, the various network structures that could be generated from the decisions of the first network had to be constructed, coded, and analyzed. Finally, the impact from the integration of each of these modifications had to be assessed.

The CIRF repair option is based on a generalized interpretation of how the CIRF maintenance concept is

currently employed by the majority of the system program offices. Extension of the existing NRLA assumptions were made with one exception. Presently, the NRLA model assumes an uniform distribution for the basing of the total number of weapons systems being analyzed. However, in the CIRF scenario, there are additional deployment factors to be considered. Although the number of systems at each base is the same, the overriding question is the number of CIRF locations and the number of bases that are being serviced by that repair location. As an illustration, in the common situation where there are two centralized servicing facilities, one in the CONUS and one overseas, the number of bases linked to each CIRF can create an imbalanced workload. In an attempt to compensate for this imbalance, it was decided that an existing systems variable called 'OS', the fraction of the force overseas, would be applied to SE utilization requirements. The application of this variable limits the number of CIRF locations to two; one overseas and one in the CONUS. It is applied by taking the standard CIRF factors and multiplying them by the overseas fraction. These numbers are then used to calculate the SE requirements for the overseas CIRF location. The same logic is applied to the CONUS CIRF SE requirements, only now the multiplication factor is $(1-OS)$. This results in a more accurate assessment of the true SE requirements for the CIRF alternative.

Input Data Requirements

To generate a CIRF repair option a list of the

necessary CIRF variables was compiled. Once this had been done, the appropriate READ and WRITE statements were located in the program and modified to accept these additional variables. A list of these new variables and their meaning is contained in Appendix A. One particular situation that occurred during this process related to the changes that had to be made to accomodate the CIRF support equipment. Currently, the NRLA program uses the first digit of the value stored in the SECODE array to identify both the type of SE and the level at which it is used. The digits and their meaning are as follows:

- 1 - Common SE (Depot)
- 2 - Peculiar SE (Depot)
- 3 - Add'l SE (Depot)
- 4 - SE Software (Depot)
- 5 - Common SE (Base)
- 6 - Peculiar SE (Base)
- 7 - Add'l SE (Base)
- 8 - SE Software (Base)

Since the SECODE array is formatted for a four digit integer value this allowed a user to input up to 999 different types of SE in any of these classifications. To differentiate between the CIRF SE and the depot or base SE, and to also maintain the capability to have a maximum of 999 pieces of SE, the SECODE values were expanded to five digit numbers. The first two digits of the SECODE value were used for the CIRF SE as follows:

- 91 - Common SE (CIRF)
- 92 - Peculiar SE (CIRF)
- 93 - Add'l SE (CIRF)
- 94 - SE Software (CIRF)

Thus a piece of support equipment that was previously identified with the number 2013 will now be identified as 20013. If this piece of equipment needed to be used at the CIRF location its identification number would be 92013.

Cost Requirements

Each of the NRLA cost subroutines (FMCMP, LRUCMP, and SRUCMP) were expanded to incorporate a capability to compute CIRF related pipeline and SE costs. The CIRF equations were developed to be compatible with the existing depot, base, and scrap cost equations. Using the CIRF input variables, the CIRF pipeline costs were calculated based on the eleven logistical costs in Table I on page 8. Two additional cost arrays, CIRFLC and CIRFSC, were needed to store these costs for LRUs and SRUs, respectively.

To compute the CIRF SE costs, the SE requirements for each of the CIRF locations were separately determined. These requirements were then added together to obtain the total needs of the system. This value was then multiplied times the unit cost of the SE (SE array CADB) to obtain the total CIRF acquisition costs. The standard cost factor for operations and maintenance contained in SE array CODB was added to the acquisition cost along with any costs that were known to be incurred for special facilities (SE array FDB). To obtain a per base cost the summation of these costs was then divided by the total number of bases in the system. These changes are contained in subroutine SECMP in Appendix H. Having calculated the costs for a CIRF repair option, it was now

necessary to design the correct RLA networks for the CIRF option.

CIRF Network Conceptual Development

Initially, the incorporation of CIRF into the NRLA model was visualized as requiring only a simple modification of the network structure to include the CIRF option. However, this proved to be false. A major restriction on this modification was that it had to be accomplished without losing the ability to identify the optimal repair level decisions through the min-cut set. Upon investigation, it became apparent that the inclusion of CIRF in this manner could not be accomplished. There is no network structure that retains the ability to identify a unique repair level decision from the four repair options. This is because of the unique use of a maximal flow algorithm to solve the minimal cost problem.

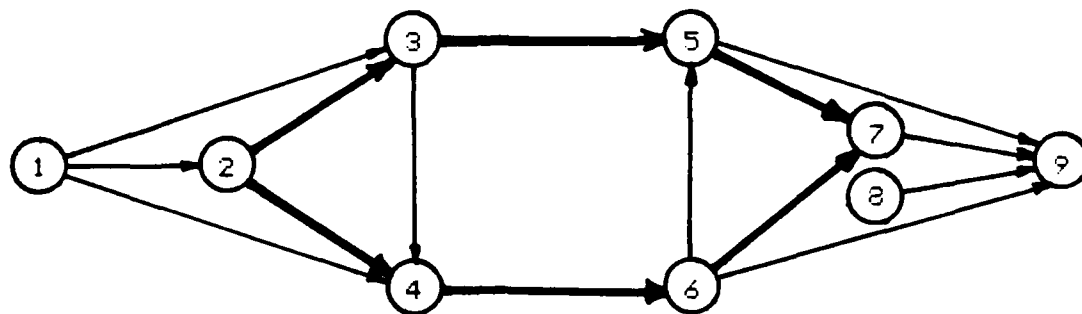
The next option considered, involved exploring the feasibility of making the decisions in a two-pass scenario. The initial pass would compare the standard depot-scrap-base configuration and the second pass would then compare these optimal decisions against the CIRF option. This method of incorporating CIRF proved to be feasible and was developed for inclusion in NRLA.

An initial problem to be overcome with this method was how to handle the seven potential SRU/LRU decision pairs which can result from the first pass. In each instance the network generating subroutine must identify the SRU/LRU

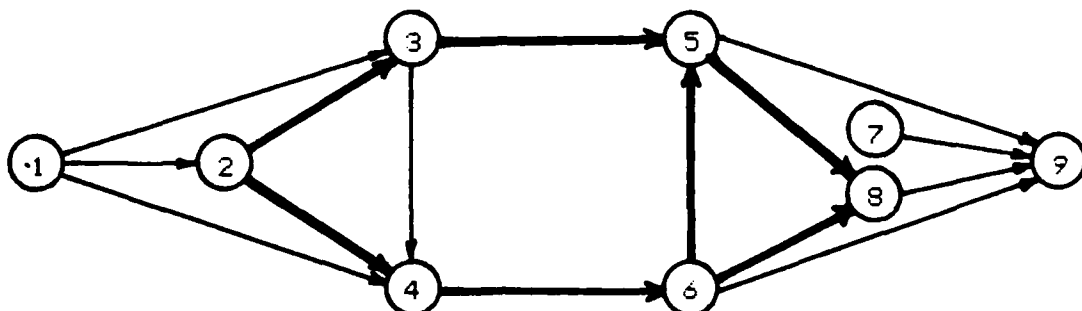
decision pair and build an appropriate network structure so that the resulting comparison with CIRF will provide valid repair level decisions. The potential for seven differently constructed subnetworks brings up the problem of how to identify the optimal decisions in these different structures. The solution to these problems led to the development of the set of seven subnetworks shown in Figures 13(a) thru 13(g). As can be seen in these figures, the basic RLA network structure is retained for each subnetwork. One major difference is that both the depot and base SE are now included on the right-hand side of the network structures.

CIRF Network Construction. Constructing the networks displayed in Figure 13 required a major modification of the SETNET subroutine. The manner in which the nodes were assigned changed very little. The source is assigned first, followed by the CIRF SE, then the CIRF LRUs and SRUs. These nodes replace the depot equivalents in the original RLA networks. The next nodes assigned are the LRU and SRU nodes associated with the optimal decisions from the first NRLA pass. These nodes take the place of the base LRU and SRU nodes in the original RLA networks. Next the depot SE nodes are assigned, followed by the base SE nodes, and then the sink node. This structure requires that the total number of nodes be increased by the number of CIRF SE.

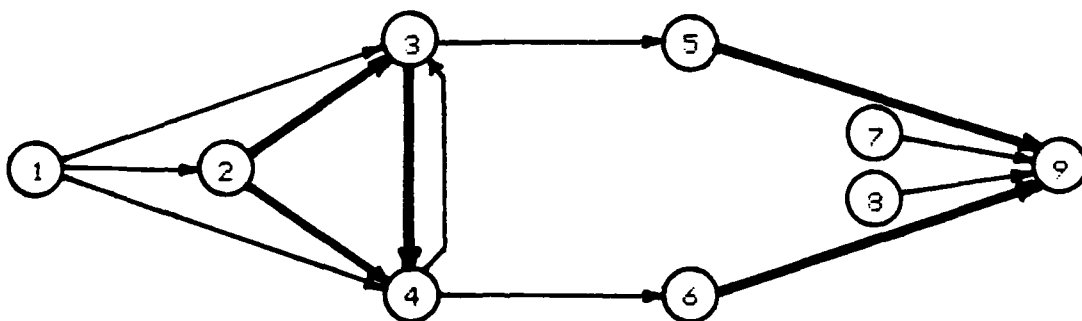
The manner in which the arcs are connected to these nodes is the key ingredient in obtaining the optimal repair level decisions. The arcs originating at the source and CIRF SE



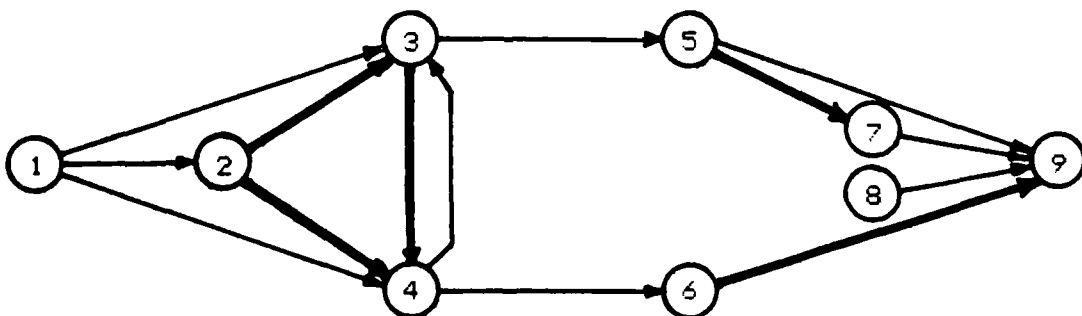
(a) Depot/Depot



(b) Base/Base

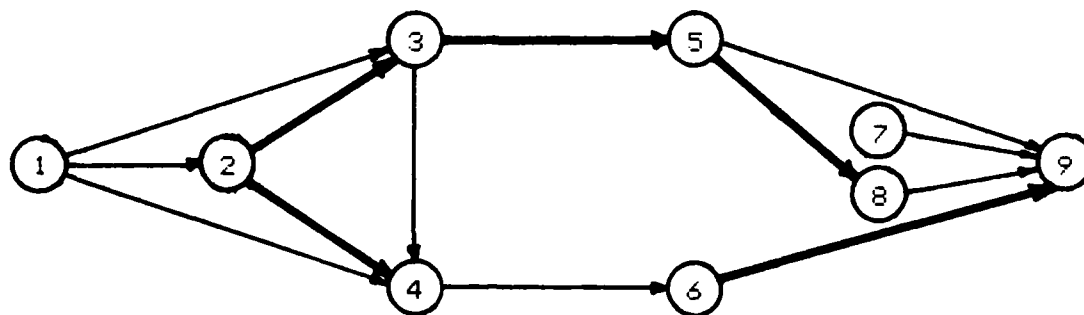


(c) Scrap/Scrap

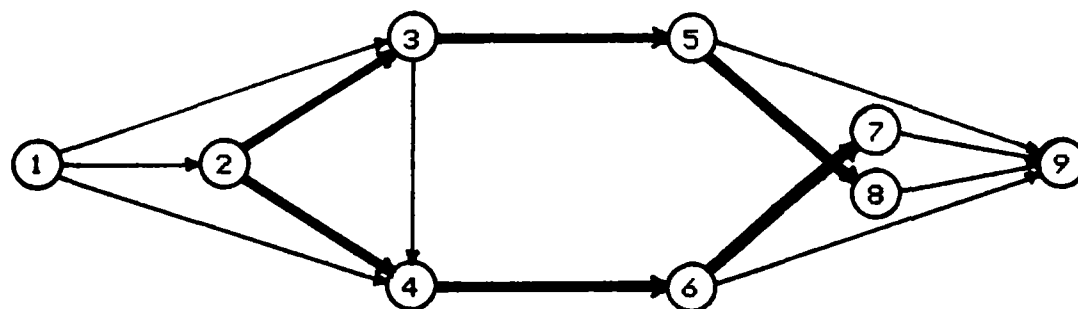


(d) Depot/Scrap

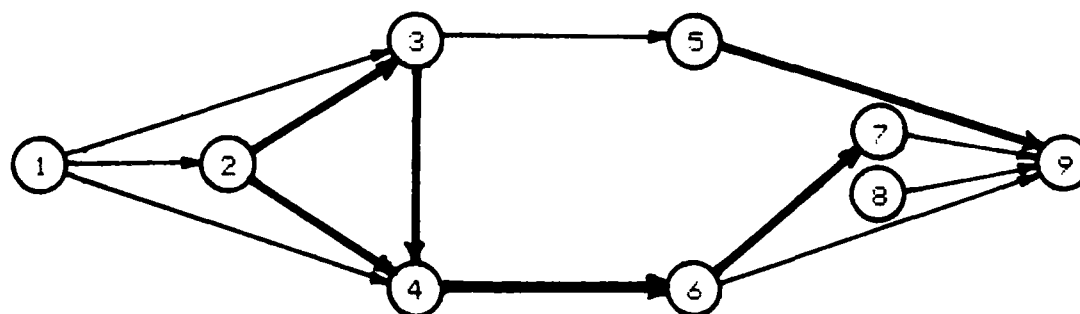
Figure 13 (a,b,c,d) CIRF RLA Subnetworks



(e) Base/Scrap



(f) Base/Depot



(g) Scrap/Depot

Figure 13 (e,f,g) CIRF RLA Subnetworks

are the same as those in the original RLA network. The first arcs requiring special attention are those that originate at the CIRF LRUs.

The arcs originating from the CIRF LRUs are assigned in the following order. First the arc connecting the LRU to the associated CIRF SRU, and then the arc connecting the LRU to the NRLA decision LRU. The first arcs capacity will be DEC 3 if the LRU decision from the first pass is base repair, and JUMBO otherwise. The DEC 3 costs are those pipeline costs associated with base repair of the LRU and CIRF repair of the SRU. The JUMBO capacity is used to exclude undesirable decisions such as scrapping or depot repairing the LRU and CIRF repairing the SRU. The second arc is associated with the scrap decision for the LRU and as such will have a capacity of JUMBO if the first pass decision was not scrap. If the decision was scrap, then the capacity of this arc is the cost associated with the scrap decision.

The CIRF SRU arcs are assigned next. The first arc connects the SRU to the CIRF LRU node. The capacity of this arc is DEC 2 and it corresponds to the pipeline costs of replacing the SRU when the LRU is CIRF repaired and the SRU is scrapped. The next arc originating from the CIRF SRU is connected to the NRLA SRU decision node. As with the LRU above, this arc's capacity will be JUMBO if the SRU decision was not scrap. If the decision was scrap, then a check is made to determine which decision was made for the LRU. If the LRU decision was base repair, then the DEC 2 costs are

added to the SRU scrap cost and this is used as the arc capacity. If the LRU decision was not base, then the SRU scrap cost alone is used as the arcs capacity.

The next arcs entered are those associated with the NRLA LRU decisions. The first arc connects the LRU node to the sink. If the NRLA decision was scrap, then the capacity of this arc is JUMBO. If it was base or depot, then the corresponding LRU repair cost is used as its capacity. Following this, the JUMBO capacity arcs connecting the LRU to the SE are assigned. The only arcs assigned, connect the LRU to the SE associated with the NRLA LRU repair decisions. As an example, if the LRU decision was base repair, then the arcs connecting the LRU node to the base SE would be assigned.

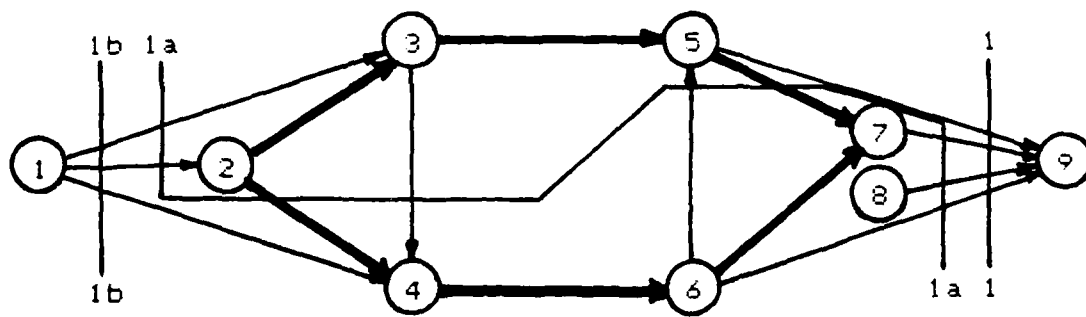
The NRLA SRU arcs are assigned next. The first arc assigned connects the NRLA SRU to the NRLA LRU. This arc is assigned in only two instances; (1) if the NRLA decisions were base repair of both the SRU and LRU and (2) if the NRLA decisions were depot repair of both. When the decisions are base repair, the arc capacity is JUMBO. This ensures that the SRU is not base repaired when the LRU is CIRF repaired. When the decision is depot repair of both, then the arc capacity is the sum of the DEC 1 and DEC 2 costs. This adds the appropriate DEC costs for CIRF LRU repair and depot SRU repair. Following this, the arcs to the sink are added. The capacity of these arcs are JUMBO if the NRLA decision was scrap the SRU. If both the LRU and SRU are repaired at the same facility the corresponding SRU repair cost is used as the capacity for this arc. In the instances where the LRU is

decisions are scrap or base repair and the SRU decision is depot repair, then the arc capacity is the sum of the SRU depot repair cost plus the DEC 1 and DEC 2 costs. The last arcs added that originate from the NRLA SRUs are the JUMBO capacity arcs to the appropriate SE resources.

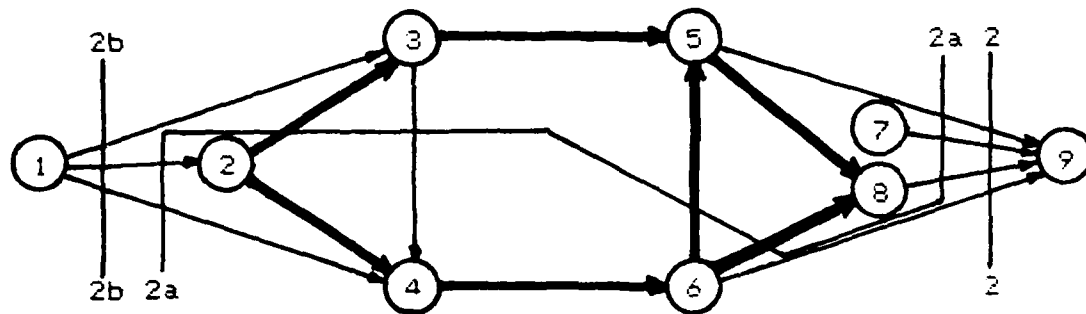
The first arcs added to the network connect the base and depot SE resources to the sink. The capacity of these arcs are the SE costs.

Although the above set of network structures is fairly complicated, it does provide a unique set of repair level decisions for the DEPOT-SCRAP-BASE-CIRF repair options. A vital element of this structure is the standardization of the CIRF repair nodes and arcs on the left side of the network. This allows the output subroutine to identify changes in the repair level decisions by just checking the state of the CIRF LRU and SRU nodes. If these nodes are in state 1 (unlabeled), then the LRU or SRU optimal repair level decision is CIRF. If the nodes are in state 2 (labeled) then the original NRLA decision is optimal.

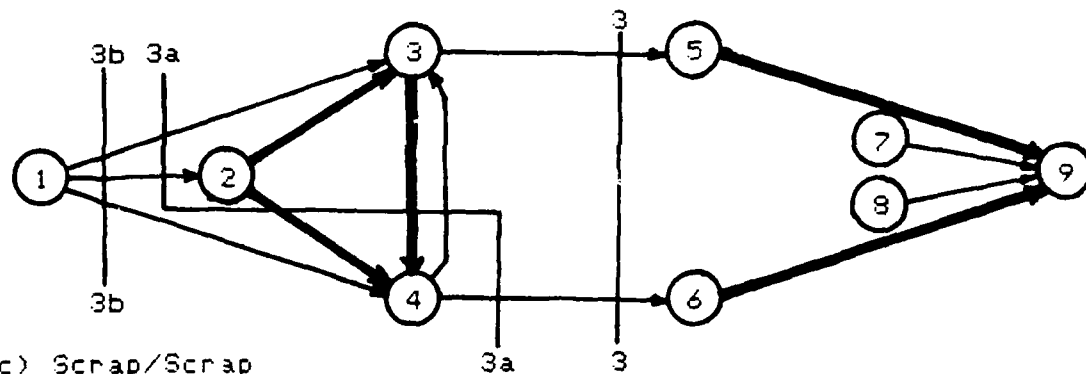
CIRF Network Cut Sets. The differences in the structures of the subnetworks lead to different cut-sets for each structure. Figures 14(a) thru 14(g) depict the feasible cut-sets possible. Table V presents the costs associated with these cuts. The same restrictions apply to these cut sets that applied to the original NRLA cut sets. The JUMBO capacity arcs cannot be part of the cut set, and the cut arcs must originate at a labeled node and terminate at an unlabeled one.



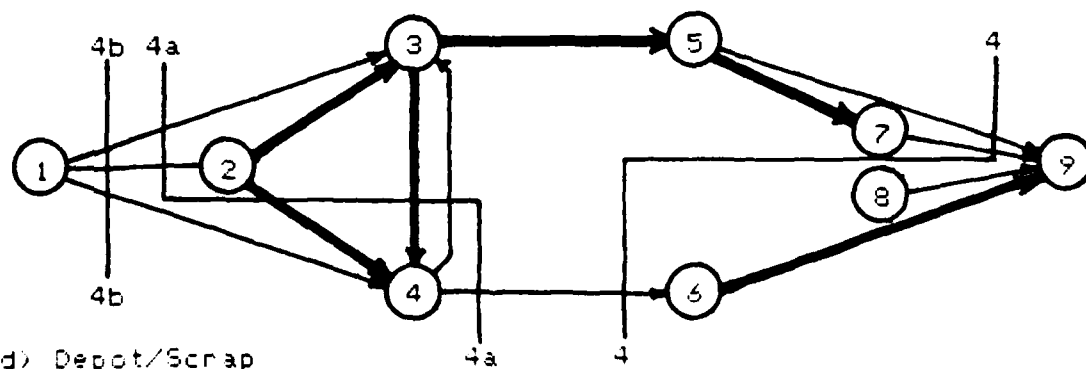
(a) Depot/Depot



(b) Base/Base

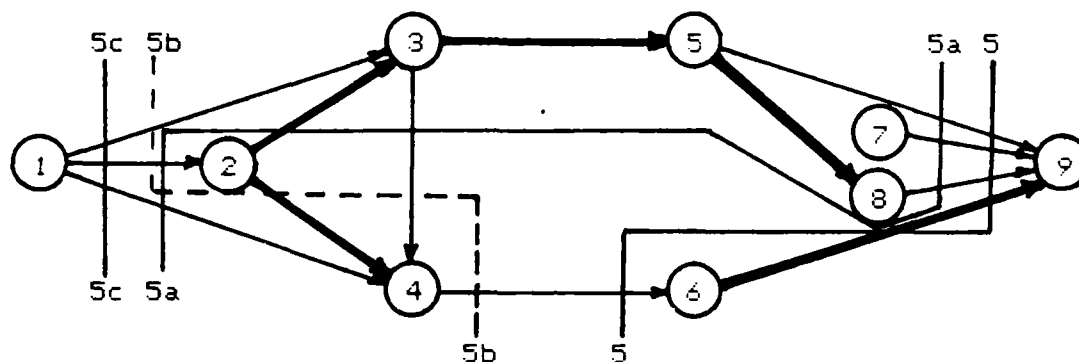


(c) Scrap/Scrap

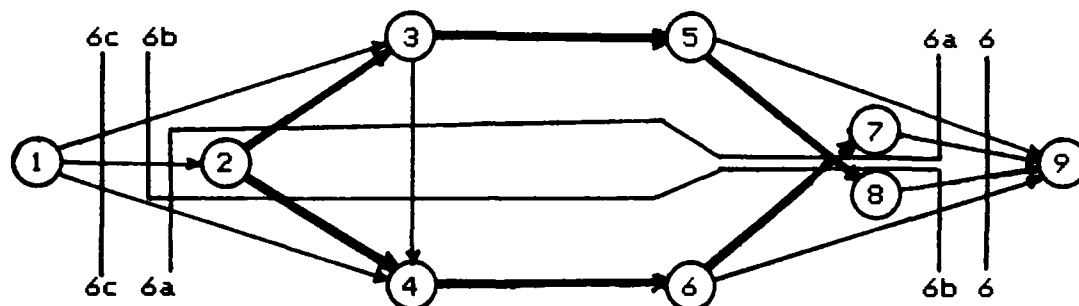


(d) Depot/Scrap

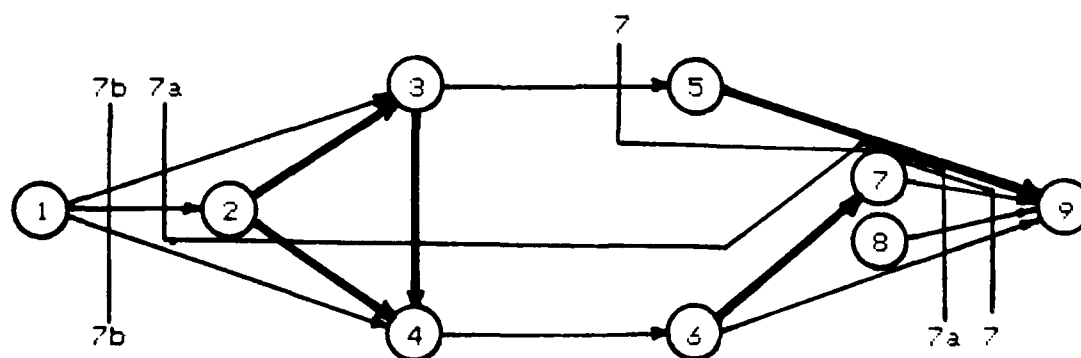
Figure 14 (a,b,c,d) CIRF RLA Network Potential Cuts



(e) Base/Scrap



(f) Base/Depot



(g) Scrap/Depot

Figure 14 (a,f,g) CIRF PLA Network Potential Cuts

TABLE VI

CIRF Network Decision Costs								
	Decision		Included Costs					
	LRU	SRU	DEP	SE BASE	CIRF	DEC 1	DEC 2	DEC 3
1	Depot	Depot	X					
1.a	CIRF	Depot	X		X	X	X	
1.b	CIRF	CIRF			X			
2	Base	Base		X				
2.a	Base	CIRF		X	X			X
2.b	CIRF	CIRF			X			
3	Scrap	Scrap						
3.a	CIRF	Scrap			X		X	
3.b	CIRF	CIRF			X			
4	Depot	Scrap		X				
4.a	CIRF	Scrap			X		X	
4.b	CIRF	CIRF			X			
5	Base	Scrap		X			X	
5.a	Base	CIRF		X	X			X
5.b	CIRF	Scrap			X		X	
5.c	CIRF	CIRF			X			
6	Base	Depot	X	X		X	X	
6.a	Base	CIRF		X	X			X
6.b	CIRF	Depot	X		X	X	X	
6.c	CIRF	CIRF			X			
7	Scrap	Depot	X			X		
7.a	CIRF	Depot	X		X	X	X	
7.b	CIRF	CIRF			X			

Computerization/Integration

The inclusion of a CIRF capability in the NRLA program required extensive changes to the program. Subroutines OUTPUT and SECMP were modified to accept and work with the new SE codes. These subroutines also identify when CIRF is included in the analysis so that the appropriate calculations can be performed. The extensive reports prepared by NRLA were a significant hinderance in achieving complete program integration. The code listing for OUTPUT and SECMP are contained in Appendixes G and H. Subroutine CRFSET creates the new network structures and a listing of it is contained in Appendix C. Once again all changes and code were accomplished using FORTRAN IV for compatibility. In addition to the above major changes, numerous smaller changes were made in subroutines LRUCMP, FMCMP, and SRUCMP. The MAIN program also required changes to incorporate CIRF. This portion of the program also has a significant report writing capability which required modification.

Verification/Validation

The verification of subroutine CRFSET, which builds the CIRF networks, was accomplished in two stages. The first stage involved the development of a set of network data which required the subroutine to build all possible network structures. The second stage of the verification required running the CRFSET subroutine in conjunction with SETNET and NMXFLO to ensure that the input/output interphases worked as expected and to further identify any potential problems.

AD-A141 198

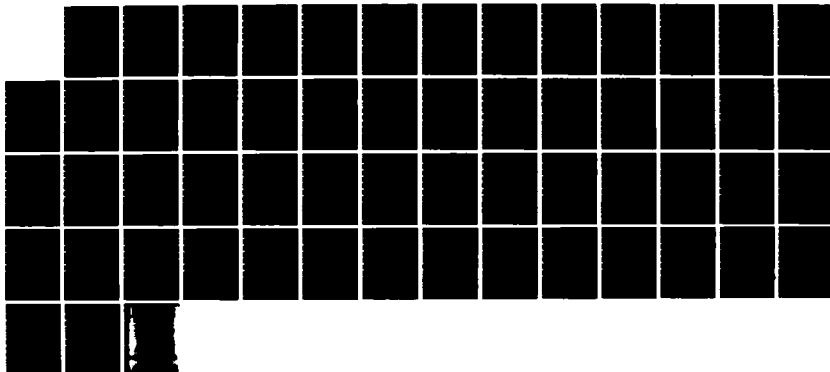
ENHANCEMENTS TO THE NETWORK REPAIR LEVEL ANALYSIS
(NRLA) MODEL USING MARG. (U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.

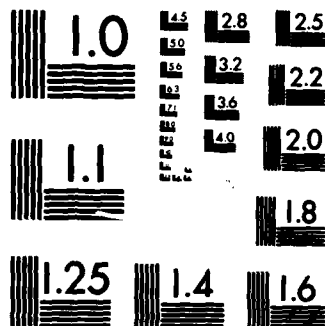
2/2

UNCLASSIFIED

G W ARNETT ET AL. DEC 83 AFIT/GOR/OS/83D-3 F/G 15/5

NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

The data set developed for this test included all possible cut set combinations so that all potential logic errors could be identified.

The data set developed for the first stage was designed to build all seven possible network structures. The arcs in these seven structures were assigned capacities equal to their arc number. The arc numbers were based on the expected arc assignments from the developed program logic. This set was then read into a test program and CRFSET was called and implemented. The resulting network structures were then compared with the expected results for accuracy of execution. When it was verified that the program was building the networks correctly the second stage of testing began.

The second stage required three data sets to test the entire series of cut set combinations. During this test the data sets were read into the test program, and processed by SETNET to build the original RLA network. This network was then solved by NMXFLO and the results printed out and checked to ensure that the input data set was producing the expected set of repair level decisions for input to CRFSET. Following this, CRFSET was called to build the CIRF network structures. These were then input to NMXFLO for solution. The results of the network max-flow solution was printed out and compared with the projected results. The results of these tests conclusively proved that subroutine CRFSET was processing the input data and producing the correct network structure in all cases. The final task dealing with the CIRF capability, was to integrate it into the NRLA program. This

task proved to be much more complicated than initially anticipated.

There were two primary factors which increased the complexity of the integration project. The first factor was the extensive work required to update the report capabilities of the program to include CIRF. This included changing much of the logic of the code so that the new SE codes could be accepted. The second factor which is directly tied to this effort is the user friendliness of the CREATE diagnostic system. On several occasions during the debugging of the program it was noted that the diagnostic outputs from the CREATE system shed little light on the nature of the problem encountered. Logic error reports were usually obscure or non-existent. In one instance an entire subroutine was missing from the program being tested, yet no recognizable diagnostic error showed up to indicate this deficiency.

Validation of the CIRF repair level option requires final integration of CIRF into the NRLA model. In addition, the cost equations developed for CIRF repair must be validated by the appropriate cost analysts in AFALC.

Analysis/Results

The final analysis of the CIRF capability requires total integration into the model and validation of the cost equations developed. The capability to include CIRF in the RLA networks is completely verified and will provide a significant analytical tool for logistics analysts when it has been integrated into the NRLA model.

VII. Conclusions/Recommendations

General Comments.

The NRLA program is a unique application of networking to solve the repair level analysis problem. It is extensively used by the Air Force logistics community to aid in determining the optimal repair level decisions for the components of new systems being procured by the Air Force. It is anticipated that models of this type will continue to play an increasingly important role in this process and as such, improvements to these models provide a significant benefit to the Air Force.

During the course of this thesis effort several areas of the NRLA program were identified as being candidates for further study and modification. In general, it is felt that the current implementation of the program should be rewritten using a more current version of FORTRAN. This will provide significant benefits in the versatility of the programming techniques available for further enhancements to the model. The sensitivity analysis section of the model requires study to determine if it can be implemented with the CIRF capability. The data structures of the model provide an area for future enhancements using new and more efficient techniques such as those found in Jensen and Barnes (4). Modification of the data structures will require a major revision of the model and, therefore, should provide an excellent opportunity to upgrade the model to the newer version of FORTRAN as recommended above.

Recommendations

MARLA. Although the MARLA enhancement has been verified and integrated into the model, conflicting repair decisions were experienced. It appears that these inconsistencies can be resolved with further effort. Based on the possible efficiencies that could be realized from the use of MARLA as a preprocessor to the NRLA model, it is recommended that this effort be undertaken. This recommendation is compatible with the Air Force guidance as outlined in AFLC/AFSC Pamphlet 800-4 (13). Following this, MARLA should be fully tested by the users to determine the extent of the benefits available from its use. For very large systems MARLA may be the best tool available to reduce the size of the data input to NRLA. In any case MARLA can be utilized as a separate program to perform marginal analysis by developing it as a program for use on micro computer systems.

Labeling Improvements. The Depth First Search Labeling algorithm developed for use in the NMXFLO subroutine is an original variation of currently accepted networking algorithms. Based on the results achieved during this study a high level of confidence was developed in its ability to outperform the existing procedure as well as other alternative methods. From this perspective it is recommended that this enhancement be immediately implemented for use in the NRLA model and distributed to current users of the model as a modular replacement for the MAXFLO subroutine. The use of this subroutine should significantly reduce the computer

resources expended during NRLA analysis, thus saving the Air Force time and money.

CIRF The ability to analyze a CIRF repair option is a sorely needed capability in the logistics planning area. By using a new series of network structures an answer has been formulated to this problem. Verification of this effort has been completed and full integration into the NRLA program is recommended at this time. A special consideration related to the CIRF effort is the validation of the cost equations by qualified cost analysts in AFALC.

Conclusion.

In summary, for the integrated logistics support process to be successful, requires that repair level analysis be skillfully performed. Without question, the problem solving technique employed in the NRLA model produces economically based optimal repair level recommendations. However, it should be remembered that the validity of these computer generated decisions is directly dependent upon the accuracy of the data input and the cost estimating relationships used.

By accomplishing the stated thesis objectives, substantial progress has been achieved in increasing both the efficiency and the capabilities of the NRLA model. Because of these improvements, it is now appropriate to investigate methods to incorporate the other non-economic factors which influence the repair level decision. A possible approach could be the development of an objective function consisting of the elements (economic and non-economic) which are

identified as critical to the systems performance. This functional relationship could then be used to establish the appropriate "cost" for each arc's capacity in the network. In this way such factors as a system's mobility and deployment requirements, availability of skilled maintenance personnel, or operational readiness requirements could be considered as part of the analytical optimization process used by NRLA. Proper integration of these effects would better reflect the total dynamics of the multi-item, multi-indenture, multi-level system which has been studied during this effort. If these recommendations are implemented a more responsive and versatile tool will be available to Air Force decision makers in the planning and management of the logistical support structures necessary for tomorrow's weapons systems.

Appendix A: Glossary of Variables

- CAA - available work time per month for a centralized intermediate level maintenance man (man-hours/month, Maintenance System Data Record).
- CIRFLC - an array of values, one for each LRU failure mode, each of which is the sum of certain life cycle logistics costs associated with intermediate level repair of the LRU.
- CIRFSC - an array of values, one for each SRU, each of which is the sum of certain life cycle logistics costs associated with intermediate level repair of the SRU.
- CLR - hourly labor rate for centralized intermediate level maintenance man (\$/hour, Maintenance System Data Record).
- CMMH - the number of maintenance man-hours required for repair of an LRU if the repair is done at CIRF level (man-hours/repair, LRU Failure Mode Data Record).
- CMMHS - the number of maintenance man-hours required for repair of an SRU if the repair is done at the CIRF level (man-hours/repair, LRU Failure Mode Data Record.)
- CRCTC - the elapsed time from removal of a failed LRU at CONUS CIRF until the item could become a serviceable spare in depot stock, it includes the time required for base to depot transportation and the depot shop flow time required for repair (months, LRU Data Record).
- CRCTO - the elapsed time from removal of a failed LRU at an overseas CIRF until the item could become a serviceable spare in depot stock (months, LRU Data Record).
- CRCTPL - the expected number of unserviceable LRU assets in the CIRF repair pipeline (No.LRU,Computed).
- CRCTSL - the number of spare LRUs to be purchased to satisfy LRU demands expected to occur during the CIRF repair cycle time (No. LRUs,Computed).
- HRAUSE - an array for each item of support equipment which stores the available equipment hours.

- NAPATH - an array which is used to store the arcs on the path from the source to the sink.
- NTCLFM - an array of values, one for each LRU failure mode, each of which is the minimum number of CIRF level maintenance personnel to be trained for the LRU repair task (number of people, LRU Failure Mode Data Record).
- NTCS - an array of values, one for each SRU, each of which is the minimum number of CIRF level maintenance personnel to be trained for the SRU repair task (number of people, SRU Data Record)
- SCRCTC - SRU CIRF repair cycle time for CONUS bases; the elapsed time from removal of a failed SRU (from the LRU) at a CONUS base until the item could become a serviceable spare in depot stock (months, SRU Data Records).
- SCRCTL - the number of spare SRUs to be purchased to satisfy SRU demands expected to occur during the SRU CIRF repair cycle time (No.SRU,Computed).
- SCRCTO - SRU CIRF repair cycle time for overseas bases; the elapsed time from removal of a failed SRU from the LRU at an overseas base until the item could become a serviceable spare in depot stock (months,SRU Data Record).
- SCRCTP - the expected number of unserviceable SRU assets in the CIRF repair pipeline (No.SRU,Computed).
- SEARY - a support equipment/LRU/SRU cross reference array, which stores the pointers of the SE for a particular failure mode or SRU.
- TBCST - a cost array which collects the total cost of repair at the base level for an LRU or SRU.
- TCCST - a cost array which collects the total cost of repair at the CIRF for an LRU or SRU.
- TDCST - a cost array which collects the total cost of repair at the depot level for an LRU or SRU.
- TFD - training factor for depot; the expected number of times that formal maintenance training will be required for depot personnel (dimensionless, Computed).
- TLCD - total life cycle repair demands for an LRU at each base (No.repair demands,/L.C.,Computed).

- TLCDF - total life cycle repair demands for a particular failure mode of an LRU at each base (No. repair demands, /L.C., Computed).
- TRB - annual turnover rate for intermediate level maintenance personnel (fraction of personnel replaced/year, Maintenance System Data Record).
- TRC - the expected training cost, instruction and materials, for LRU repairs (\$/Man/week, LRU Failure Mode Data Record).
- TSECB - a cost array used to collect the SE costs at the base level for a particular LRU or SRU.
- TSECC - a cost array used to collect the SE costs at the CIRF for a particular LRU or SRU.
- TSECD - a cost array used to collect the SE costs at the depot level for a particular LRU or SRU.
- TSECST - the total cost associated with the acquisition and operation of a particular piece of support equipment.

Appendix B: NRLA Subroutine Descriptions

- DECIDE - this subroutine is called from UCLSA and MTBFSA. Its purpose is to specifically determine the decision changes identified by the sensitivity analysis routines and document the changes on an output file.
- FMCMP - this subroutine is called by MAIN, MTBFSA, and UCLSA to compute repair level option costs for an LRU failure mode. These costs are saved in arrays DEPOLC, SCRPLC, CIRFLC, and BASELC. The expected number of maintenance man-hours required monthly for the failure mode is computed for depot, CIRF, and base and saved in FSEUHD, FSEUHC, and FSEUHB respectively.
- LRUCMP - this subroutine is called by MAIN, MTBFSA and UCLSA. Its function is to compute inventory stock levels and life cycle SCRAP option costs for an LRU.
- MAXFLO - this subroutine is called by MAIN, MTBFSA, and UCLSA. Its purpose is to determine the maximum flow through the network and the minimum cut set. This cut set identifies the minimum cost set of repair level decisions.
- MTBFSA - this subroutine is called from MAIN to determine the effects of changes in the MTBF for an LRU. The effects could be changes in failure mode repair level decisions, SRU repair level decisions, and/or changes to SE decisions.
- OUTPUT - this subroutine is called from MAIN to print the optimal solution results. This includes support equipment decisions plus LRU and SRU repair level decisions.
- RESET - this subroutine is called by MAIN, MTBFSA, and UCLSA. Its purpose is to exploit the known structure of the RLA network so that advance flow can be placed on the network arcs to reduce the solution time for the MAXFLO subroutine.
- SECMP - this subroutine is called by MAIN and MTBFSA. It determines the quantity of each SE resource potentially required at depot, CIRF, and base level plus the life cycle cost for the resources.

- SETNET - this subroutine is called by MAIN after all costs and dependency relationships are known. The function of SETNET is to use this information to construct a matrix representation of the RLA network, determine and save pointers for use by the solution algorithm and save pointers for use by the sensitivity analysis subroutines.
- SORT - this subroutine is called from SETNET. As its name implies, its function is to sort values supplied by SETNET. Specifications to SORT are contained in its arguments list.
- SRUCMP - this subroutine is called by MAIN, MTBFSA, and UCLSA to compute SRU related life cycle costs. These costs are saved in the SRU related arrays DEPOSC, SCRPSC, CIRFSC, BASESC, BRLSS, BRLDS, and BRLCS.
- UCLSA - this subroutine is called by MAIN to determine the effects of LRU and SRU unit cost changes. The routine determines if decision changes will occur over a range of values for both the LRU's and SRU's unit cost.

Appendix C: CRFSET Subroutine

```

10      SUBROUTINE CRFSET(MSG12)
20C
30C
40      ICIRF=2
50C
60C      CODE FROM STATEMENT # 2000 THROUGH # 2210 UTILIZES THE ABOVE
70C      COMPUTED ITEM & SE COSTS TO PRODUCE AN RLA COST NETWORK.
80C
90C      AS A PRELIMINARY STEP DETERMINE NODE NUMBERS FOR KEY NODES IN
100C     THE NETWORK. THE VARIABLES ARE CODED WITH THE PREFIXES 'LC'
110C     FOR 'LAST CIRF' AND 'LN' FOR 'LAST NRLA INPUT'
120C
130 2000 LCSE=NCSE+1
140      LCLRU=LCSE+LFMS
150      LCSRU=LCLRU+SFMS
160      LNLRU=LCSRU+LFMS
170      LNSRU=LNLRU+SFMS
180      LNDSE=LNSRU+NDSE
190      LNBSE=LNDSE+NBSE
200      LNODE=LNBSE+1
210      IF(LNODE.LE.MAXNOD) GO TO 2005
220      WRITE(6,2002)
230 2002 FORMAT('0 NODE VECTORS ARE TOO SMALL -- STOP')
240      STOP
250C
260C
270C      SET # OF ARCS TO ZERO & # OF NODES TO 1
280 2005 NARCS=0
290      NNODES=1
300      NBSEP0=NDSE+NBSE
310      NBSFP1=NDSE+NBSE+1
320C
330      IF(NCSE.LT.1) GO TO 2015
340C      CREATE ARCS FROM SOURCE TO CIRF SE -- CIRF SE COST ARCS
350      DO 2010 I2010=NBSFP1,NUMSER
360      NARCS=NARCS+1
370      NNODES=NNODES+1
380      SRCE(NARCS)=1
390      DEST(NARCS)=NNODES
400      CAP(NARCS)=SECCOST(I2010)
410 2010 CONTINUE
420C
430C      CREATE ARCS FROM THE SOURCE TO THE CIRF LRU FAILURE MODE
440C      NODES -- ITEM RELATED COSTS FOR LRU REPAIR AT CIRF
450 2015 DO 2020 I2020=1,LFMS
460      NARCS=NARCS+1
470      NNODES=NNODES+1
480      SRCE(NARCS)=1
490      DEST(NARCS)=NNODES
500      CAP(NARCS)=CIRFIC(I2020)

```



```

510 2020 CONTINUE
520C
530C     CREATE ARCS FROM THE SOURCE TO THE CIPF SRU NODES -- ITEM
540C     RELATED COSTS FOR SRU REPAIR AT CIPF
550     IF(SFMS.LT.1) GO TO 2040
560     DO 2030 I2030=1,SFMS
570     NARCS=NARCS+1
580     NNODES=NNODES+1
590     SRCE(NARCS)=1
600     DEST(NARCS)=NNODES
610     CAP(NARCS)=CIRFSC(I2030)
620 2030 CONTINUE
630C
640C     CREATE ARCS FROM THE CIPF SE NODES TO THE CIPF LRU/SRU NODES
650 2040 IF(NCSE.LT.1) GO TO 2070
660C     FOR EACH CIPF SE --
670     DO 2060 I2060=NBSEP1,NUMBER
680C     SEPF CONTAINS A POINTER TO THE SE CROSS REFERENCE (SEXREF)
690     IPTR=SEPF(I2060)
700     IF(IPTR.EQ.0) GO TO 2060
710C     SEXREF CONTAINS THE # OF AN LRU FAILURE MODE (ITEM > 0) OR
720C     THE NUMBER OF AN SRU (ITEM < 0)
730 2050 ITEM=SEXREF(IPTR)
740     IF(ITEM.LT.0) ITEM=LFMS-ITEM
750     NARCS=NARCS+1
760     SRCE(NARCS)=I2060-NBSEP0+1
770C     ITEM IS A POINTER TO AN LRU OR SRU AND WHEN ADDED TO LCSE
780C     GIVES THE APPROPRIATE NODE NUMBER
790     DEST(NARCS)=LCSE+ITEM
800     CAP(NARCS)=JUMBO
810C
820C     NXTITM IS A POINTER TO THE NEXT ENTRY IN SEXREF WHICH IS AN
830C     LRU OR SRU REQUIRING THE CURRENT SE, I.E., SE # I2060.
840     NPTR=NXTITM(IPTR)
850     IF(NPTR.EQ.0) GO TO 2060
860     IPTR=NPTR
870     GO TO 2050
880 2060 CONTINUE
890C
900C     CREATE THE ARCS EMANATING FROM THE CIPF LRU FAILURE MODE NODES
910 2070 DO 2080 I2080=1,LFMS
920C     CHECK FOR AN SRU
930     IPTR=SRUPTR(I2080)
940     IF(IPTR.EQ.0) GO TO 2075
950C     CREATE AN ARC TO THE CIPF SPU NODE -- COSTS UNIQUE TO BASE
960C     REPAIR OF LRU & CIPF REPAIR OF SRU
970     NARCS=NARCS+1
980     SRCE(NARCS)=LCSE+I2080
990     DEST(NARCS)=LCLRU+IPTR
1000    CAP(NARCS)=BR LCS(IPTR)
1010    IF (OPDECL(I2080).NE.LOCAT(3)) CAP(NARCS)=JUMBO
1020C     CREATE AN ARC TO THE NRLA LRU NODES -- COST OF SCRAPPING THE LRU

```

```

1030 2075 NARCS=NARCS+1
1040     SRCE(NARCS)=LCSE+I2080
1050     DEST(NARCS)=LCSRU+I2080
1060     CAP(NARCS)=SCRPLC(I2080)
1070     IF (OPDECL(I2080).NE.LOCAT(2)) CAP(NARCS)=JUMBO
1080 2080 CONTINUE
1090C
1100C     CREATE ARCS FROM THE CIRF SRU NODES TO THE NRLA SRU NODES --
1110C     COST OF SCRAPPING THE SRU
1120     IF(SFMS.LT.1) GO TO 2100
1130     DO 2090 I2090=1,SFMS
1140C     CREATE ARCS FROM THE CIRF SRU NODES TO THE CIRF LRU
1150C     NODES, TO ELIMINATE INFEASIBLE LRU/SRU REPAIR LEVEL
1160C     MATCHES SUCH AS LRU-D, SRU-C; LRU-S, SRU-C; LRU-C, SRU-B
1170     IF (OPDECS(I2090).EQ.LOCAT(2).AND.OPDECL(LRUPTR(I2090)).
1180     &EQ.LOCAT(1)) GOTO 2082
1190     IF (OPDECS(I2090).EQ.LOCAT(2).AND.OPDECL(LRUPTR(I2090)).EQ.
1200     &LOCAT(2)) GOTO 2082
1210     GOTO 2085
1220 2082 NARCS=NARCS+1
1230     SRCE(NARCS)=LCLRU+I2090
1240     DEST(NARCS)=LCSE+LRUPTR(I2090)
1250     CAP(NARCS)=BRLSS(I2090)
1260C     CREATE ARCS FROM CIRF SRU TO NRLA SRU -- SCRAP COSTS
1270 2085 NARCS=NARCS+1
1280     SRCE(NARCS)=LCLRU+I2090
1290     DEST(NARCS)=LNLRU+I2090
1300     CAP(NARCS)=SCRPSC(I2090)
1310     IF (OPDECS(I2090).NE.LOCAT(2)) CAP(NARCS)=JUMBO
1320C     ADD DEC2 COSTS TO SCRAP CSOT FOR BASE SCRAP DECISION
1330     IF (OPDECL(LRUPTR(I2090)).EQ.LOCAT(3).AND.OPDECS(I2090).EQ.
1340     &LOCAT(2)) CAP(NARCS)=CAP(NARCS)+BRLSS(I2090)
1350 2090 CONTINUE
1360C
1370C     CREATE ARCS EMANATING FROM THE NRLA LRU NODES
1380 2100 DO 2120 I2120=1,LFMS
1390C     CREATE AN ARC FROM THE NRLA LRU NODE TO THE LAST NOD (SINK) --
1400C     ITEM RELATED COST OF NRLA REPAIR OF THE LRU
1410 2110 NARCS=NARCS+1
1420     SRCE(NARCS)=LCSRU+I2120
1430     DEST(NARCS)=LNODE
1440     CAP(NARCS)=JUMBO
1450     IF (OPDECL(I2120).EQ.LOCAT(3)) CAP(NARCS)=BASELC(I2120)
1460     IF (OPDECL(I2120).EQ.LOCAT(1)) CAP(NARCS)=DEPOLC(I2120)
1470 2120 CONTINUE
1480C
1490C     CREATE ARCS EMANATING FROM THE NRLA SRU NODES
1500     IF (SFMS.LT.1) GOTO 2140
1510     DO 2130 I2130=1,SFMS
1520C     CREATE AN ARC FROM THE NRLA SRU TO NRLA LRU IF THE
1530C     NRLA DECISION IS EITHER LRU-D, SRU-D OR LRU-B, SRU-B
1540     IF (OPDECL(LRUPTR(I2130)).EQ.LOCAT(1).AND.OPDECS(I2130).EQ.

```

```

1550      &LOCAT(1)) GOTO 2132
1560      IF (OPDECL(LRUPTR(I2130)).EQ.LOCAT(3).AND.OPDECS(I2130).EQ.
1570      &LOCAT(3)) GOTO 2132
1580      GOTO 2135
1590 2132 NARCS=NARCS+1
1600      SRCE(NARCS)=LNLRU+I2130
1610      DEST(NARCS)=LCSRU+LRUPTR(I2130)
1620      CAP(NARCS)=JUMBO
1630      IF (OPDECS(I2130).EQ.LOCAT(3)) GOTO 2135
1640      CAP(NARCS)=BRLSS(I2130)+BRLDS(I2130)
1650C
1660C      CREATE AN ARC FROM THE NRLA SRU NODE TO THE LAST NODE (SINK) --
1670C      ITEM RELATED COST OF NRLA REPAIR FOR THE SRU
1680 2135 NARCS=NARCS+1
1690      SRCE(NARCS)=LNLRU+I2130
1700      DEST(NARCS)=LNODE
1710      CAP(NARCS)=JUMBO
1720      IF(OPDECS(I2130).EQ.LOCAT(3)) CAP(NARCS)=BASESC(I2130)
1730      IF (OPDECS(I2130).EQ.LOCAT(1)) CAP(NARCS)=DEPOSC(I2130)
1740      IF (OPDECL(LRUPTR(I2130)).NE.LOCAT(1).AND.OPDECS(I2130).EQ.
1750      &LOCAT(1)) CAP(NARCS)=CAP(NARCS)+BRLSS(I2130)+BRLDS(I2130)
1760 2130 CONTINUE
1770C
1780C      CREATE ARCS TO AND FROM THE NRLA SE NODES
1790C      ADD ARCS FOR THE NRLA DEPOT SE DECISIONS
1800 2140 IF (NDSE.LT.1) GOTO 2160
1810C      FOR EACH DEPOT SE --
1820      DO 2150 I2150=1,NDSE
1830      IPTR=SEPF(I2150)
1840      IF (IPTR.EQ.0) GOTO 2149
1850 2145 ITEM=SEXREF(IPTR)
1860      IF (ITEM.LT.0.AND.OPDECS(IABS(ITEM)).NE.LOCAT(1)) GOTO 2148
1870      IF (ITEM.GT.0.AND.OPDECL(ITEM).NE.LOCAT(1)) GOTO 2148
1880      IF (ITEM.LT.0) ITEM=LIMS-ITEM
1890      NARCS=NARCS+1
1900      SRCE(NARCS)=LCSRU+ITEM
1910      DEST(NARCS)=LNSRU+I2150
1920      CAP(NARCS)=JUMBO
1930 2148 NPTR=NXTITM(IPTR)
1940      IF (NPTR.EQ.0) GOTO 2149
1950      IPTR=NPTR
1960      GOTO 2145
1970C      ARCS FROM DEPOT SE TO SINK
1980 2149 NARCS=NARCS+1
1990      SRCE(NARCS)=LNSRU+I2150
2000      DEST(NARCS)=LNODE
2010      CAP(NARCS)=SEECOST(I2150)
2020 2150 CONTINUE
2030C      ADD ARCS FOR NRLA BASE DECISIONS
2040 2160 IF (NBSE.LT.1) GOTO 2180
2050C      FOR EACH BASE SE
2060      NDSEPI=NDSE+1

```

```

2070      DO 2170 I2170=NDSEPI,NESEP0
2080      IPTR=SEPF(I2170)
2090      IF (IPTR.EQ.0) GOTO 2169
2100 2165 ITEM=SEXREF(IPTR)
2110      IF (ITEM.LT.0.AND.OPDECS(LABS(ITEM)).NE.LOCAT(3)) GOTO 2168
2120      IF (ITEM.GT.0.AND.OPDECL(ITEM).NE.LOCAT(3)) GOTO 2168
2130      IF (ITEM.LT.0) ITEM=LFMS-ITEM
2140      NARCS=NARCS+1
2150      SRCE(NARCS)=LCSRU+ITEM
2160      DEST(NARCS)=LNSRU+I2170
2170      CAP(NARCS)=JUNBO
2180 2168 NPTR=NXTITN(IPTR)
2190      IF (NPTR.EQ.0) GOTO 2169
2200      IPTR=NPTR
2210      GOTO 2165
2220C      ARCS FROM BASE SE TO SINK
2230 2169 NARCS=NARCS+1
2240      SRCE(NARCS)=LNSRU+I2170
2250      DEST(NARCS)=LNODE
2260      CAP(NARCS)=SECOST(I2170)
2270 2170 CONTINUE
2280 2180 NNODES=LNODE
2290C
2300C      EXPLICITLY SORT THE SRCE-DEST-CAP TRIPLES INTO ASCENDING ORDER
2310C      BY SRCE (FLOW IS USED BY SORT TO STORE ORDERING POINTERS)
2320      CALL SORT(SRCE,DEST,CAP,FLOW,NARCS,MAXARC,1)
2330C      SAVE POINTERS FOR FORWARD SCAN IN FWDSP
2340      DO 2190 I2190=1,NARCS
2350C      UTILIZE THE VALUES IN SRCE IN REVERSE ORDER. (IF NODE '1' IS A
2360C      SOURCE FOR 'N' ARCS THEN FWDSP(1) WILL SUCCESSIVELY GET THE
2370C      VALUES: N, N-1, N-2, . . . , 2, 1. SIMILARLY, FWDSP(K) WILL
2380C      HAVE THE VALUE 'J' WHERE J CORRESPONDS TO THE FIRST OCCURENCE
2390C      OF NODE # K IN SRCE.)
2400      J2190=NARCS+1-I2190
2410      K2190=SRCE(J2190)
2420      FWDSP(K2190)=J2190
2430 2190 CONTINUE
2440      FWDSP(LNODE)=NARCS+1
2450      LNODM1=LNODE-1
2460      DO 2195 I2195=LNODM1,1
2470      J2195=LNODE-I2195
2480      IF(FWDSP(J2195).EQ.0) FWDSP(J2195)=FWDSP(J2195+1)
2490 2195 CONTINUE
2500C
2510C      USE SORT TO EXPLICITLY SEQUENCE THE DEST ENTRIES INTO ASCENDING
2520C      ORDER. THE ORDER WILL BE SPECIFIED BY THE ENTRIES IN BKPTR.
2530C      THUS, THE ENTRIES IN BACKSP ARE POINTERS TO SRCE-DEST-CAP
2540C      TRIPLES BASED ON THE DEST VALUES.
2550      CALL SORT(DEST,0,0,BKPTR,NARCS,MAXARC,0)
2560C      SAVE POINTERS FOR BACKWARD SCAN IN BACKSP
2570      BACKSP(1)=0
2580      DO 2200 I2200=1,NARCS

```

```

2590      J2200=BKPTR(I2200)
2600      K2200=DEST(J2200)
2610C      BACKSP(K) IS THE POSITION IN BKPTR WHICH HOLDS THE LAST POINTER
2620C      TO A SRCE-DEST-CAP TRIPLE HAVING NODE # K AS THE DEST. THE FIRST
2630C      POINTER TO A TRIPLE HAVING NODE # K AS THE DEST IS BKPTR(K-1)+1.
2640      BACKSP(K2200)=I2200
2650 2200 CONTINUE
2660      DO 2222 I2222=2,LNODE
2670      IF(BACKSP(I2222).EQ.0) BACKSP(I2222)=BACKSP(I2222-1)
2680 2222 CONTINUE
2690C
2700C IF(MSGL2.LT.1) GO TO 2205
2710C PRINT 2202,'SRCE ',(SRCE(I),I=1,NARCS)
2720C PRINT 2202,FORMAT(' ',A6,(' ',20I3))
2730C PRINT 2202,'DEST ',(DEST(I),I=1,NARCS)
2740C PRINT 2202,'CAP ',(CAP(I),I=1,NARCS)
2750C PRINT 2202,'FLOW ',(FLOW(I),I=1,NARCS)
2760C PRINT 2202,'BKPTR ',(BKPTR(I),I=1,NARCS)
2770C PRINT 2202,'FWDSP ',(FWDSP(I),I=1,LNODE)
2780C PRINT 2202,'BACKSP',(BACKSP(I),I=1,LNODE)
2790 2205 DO 2400 I2400=1,NARCS
2800      J=SRCE(I2400)
2810      K=DEST(I2400)
2820      IF(J.NE.1) GO TO 2230
2830      IF(K.GT.LCSE) GO TO 2210
2840      SEARCHP(K-1)=I2400
2850      GO TO 2300
2860 2210 IF(K.GT.LCLPU) GO TO 2220
2870      LDARC(K-LCSE)=I2400
2880      GO TO 2300
2890 2220 SDARC(K-LCLRU)=I2400
2900      GO TO 2300
2910 2230 IF(J.LE.LCSE) GO TO 2300
2920      IF(J.GT.LCLRU) GO TO 2250
2930      IF(K.LE.LCSRU) GO TO 2240
2940      LSARC(J-LCSE)=I2400
2950      GO TO 2300
2960 2240 SBDARC(K-LCLRU)=I2400
2970      GO TO 2300
2980 2250 IF(J.GT.LCSRU) GO TO 2260
2990      SSARC(J-LCLRU)=I2400
3000      GO TO 2300
3010 2260 IF(J.GT.LMLRU) GO TO 2280
3020      IF(K.NE.LNODE) GO TO 2270
3030      LBARC(J-LCSRU)=I2400
3040      GO TO 2300
3050 2270 IF(K.GT.LMSRU) GO TO 2300
3060      SBSARC(K-LMLRU)=I2400
3070      GO TO 2300
3080 2280 IF(J.GT.LNSRU) GO TO 2290
3090      IF(K.NE.LNODE) GO TO 2300

```

```

3100      SBARC(J-LNLRU)=I2400
3110      GO TO 2300
3120 2290 SEARCHP(J+NCSE-LNSRU)=I2400
3130 2300 FLOW(I2400)=0
3140 2400 CONTINUE
3150C IF(MSG2.LT.2) GO TO 2440
3160C DO 2410 I=1,LFMS
3170C PRINT,'LPTRS ',LDARC(I),LSARC(I),LBARC(I)
3180C 2410 CONTINUE
3190C DO 2420 I=1,SFMS
3200C PRINT,'SPTRS ',SDARC(I),SSARC(I),SBARC(I),SEBARC(I),SBSARC(I)
3210C 2420 CONTINUE
3220C DO 2430 I=1,NUMSER
3230C PRINT,'SEPTRS ',SEARCHP(I)
3240C 2430 CONTINUE
3250C 2440 CONTINUE
3260      RETURN
3270      END

```

*

Appendix D: MARLA Subroutine

```

100
20      SUBROUTINE MARLA
300
40*     THIS PROGRAM IMPLEMENTS THE CONCEPT OF MARGINAL      *
50*     ANALYSIS TO DETERMINE THE OPTIMUM REPAIR LEVEL        *
60*     FOR AN ITEM (LRU FAILURE MODE OR SRU) OF A WEAPONS     *
70*     SYSTEM. IT WILL ACT AS A PREPROCESSOR TO THE MAXFLO    *
80*     SUBROUTINE OF THE NRLA PROGRAM.                          *
900
100*    DO LOOP 10 CONDUCTS THE ANALYSIS FOR THE FIRST ITEM   *
110*    TO THE LAST ITEM OF THE SYSTEM.                          *
1200
130      TOTITM=LEMS+SFMS
140      DO 10 I=1,TOTITM
150      IF(I.GT.LEMS) GO TO 15
160      IF(LEMOD(I).NE.1) GO TO 12
170      LEMOD(I)=2
180      INDD=1000
190 12 IF(LEMOB(I).NE.1) GO TO 14
200      LEMOB(I)=2
210      INDB=10
220 14 IF(LEMOB(I).NE.1) GO TO 16
230      LEMOC(I)=2
240      INDC=1
250 16 IF(LEMOB(I).NE.1) GO TO 18
260      LEMOS(I)=2
270      INDS=100
280 18 CONTINUE
290      IF (CIRF.EQ.0) LEMOC(I)=3
3000
310*    DO LOOP 20 CHECKS EACH POSITION OF THE SE/LRU/SRU      *
320*    CROSS REFERENCE ARRAY (SEARY) FOR THE POINTER VALUES *
330*    OF THE SE NECESSARY TO REPAIR THE Ith ITEM.            *
3400
350      DO 20 I20=1,26
360      PTR=SEARY(I,I20)
370      IF(PTR.EQ.0) GO TO 20
380      ID=SECODE(PTR)/10
390      IF(LEMOD(I).GT.0.OR.ID.GE.5000) GO TO 32
400      BUY=AINT(1.0+(FSEUHD(I)/OPHRS(PTR)))
410      TSECD(I)=TSECD(I)+(BUY*CADB(PTR)+FDB(PTR)+(FSEUHD(I)/
420      & OPHRS(PTR))*CODB(PTR)*PIUP)
430 32 IF(LEMOB(I).GT.0.OR.ID.LT.5000.OR.ID.GE.9000) GO TO 34
440      BUY=AINT(1.0+(FSEUHB(I)/OPHRS(PTR)))
450      TSECB(I)=TSECB(I)+(BUY*CADE(PTR)+FDB(PTR)+((FSEUHB(I)/
460      & OPHRS(PTR))*CODB(PTR)*PIUP))
470 34 IF(LEMOB(I).GT.0.OR.ID.LT.9000) GO TO 20
480      BUY=2+AINT((FSEUHC(I)*OS)/OPHRS(PTR))+AINT((FSEUHC(I)*(1-OS))
490      & /OPHRS(PTR))
500      TSECC(I)=TSECC(I)+(BUY*CADB(PTR)+FDB(PTR)+(FSEUHC(I)/

```

```

510      & OPFRS(PTB))*CDB(PTB)*PIPB)
520      20 CONTINUE
530C
540C      NOW THE TOTAL COST FOR EACH ITEM
550*      DETERMINED (TDCST,TBCST,AND TCCST). FINALLY USING A
560*      SERIES OF IF STATEMENTS A MARGINAL ANALYSIS IS
570*      PERFORMED TO IDENTIFY IF ONE REPAIR LEVEL IS MORE
580*      ECONOMICAL THAN ANOTHER. IF THIS IS TRUE A UNIQUE
590*      INDICATOR IS SET (INDD,INDB,INDS,INDC).
600C
610      IF(TSECD(I).EQ.0) GO TO 38
620      SED=TSECD(I)/N
630      IF(TSECC(I).EQ.0) GO TO 38
640      SEC=TSECC(I)/N
650      38 TDCST(I)=DEPOLC(I)+SED
660      TBCST(I)=BASELC(I)+TSECB(I)
670      IF(CIRF.EQ.0) TCCST(I)=JUMBO
680      TCCST(I)=CIRFLC(I)+SEC
690      IF(LEMOD(I).GT.0.OR.LEMOS(I).GT.0) GO TO 80
700      IF(SCRPLC(I).LT.DEPOLC(I)) INDD=1000
710      IF(TDCST(I).LT.SCRPLC(I)) INDS=100
720      80 IF(LEMOS(I).GT.0.OR.LEMOB(I).GT.0) GO TO 82
730      IF(TBCST(I).LT.SCRPLC(I)) INDB=100
740      IF(SCRPLC(I).LT.BASELC(I)) INDB=10
750      82 IF(LEMOB(I).GT.0.OR.LEMOD(I).GT.0) GO TO 84
760      IF(TBCST(I).LT.DEPOLC(I)) INDD=1000
770      IF(TDCST(I).LT.BASELC(I)) INDB=10
780      84 IF(LEMOD(I).GT.0.OR.LEMOB(I).GT.0) GO TO 74
790      IF(TCCST(I).LT.DEPOLC(I)) INDD=1000
800      IF(TDCST(I).LT.CIRFLC(I)) INDC=1
810      74 IF(LEMOD(I).GT.0.OR.LEMOS(I).GT.0) GO TO 77
820      IF(SCRPLC(I).LT.CIRFLC(I)) INDC=1
830      IF(TCCST(I).LT.SCRPLC(I)) INDS=100
840      77 IF(LEMOD(I).GT.0.OR.LEMOB(I).GT.0) GO TO 87
850      IF(TCCST(I).LT.BASELC(I)) INDB=10
860      IF(TBCST(I).LT.CIRFLC(I)) INDC=1
870C
880C      LINES BELOW CHECK THE VALUES OF THE INDICATORS
890*      IF THE INDICATOR HAS BEEN GIVEN THE RIGHT VALUE, THE
900*      NRLA FAILURE MODE EXCLUSION ARRAY IS UPDATED TO
910*      ELIMINATE THAT REPAIR OPTION FROM FURTHER CONSIDER-
920*      ATION FOR THE Ith ITEM.
930C
940      87 IF(LEMOD(I).EQ.2) GO TO 81
950      IF(INDD.EQ.1000) LEMOD(I)=1
960      81 IF(LEMOS(I).EQ.2) GO TO 83
970      IF(INDS.EQ.100) LEMOS(I)=1
980      83 IF(LEMOB(I).EQ.2) GO TO 85
990      IF(INDB.EQ.10) LEMOB(I)=1
1000      85 IF(LEMOD(I).GE.2) GO TO 88
1010      IF(INDC.EQ.1) LEMOD(I)=1
1020      88 TYPEPR=INDD+INDS+INDB+INDC
1030C

```



```

1040*   BASED ON THE VALUE OF TYPRPR THE BELOW STATEMENTS      *
1050*   CHECK TO SEE IF 3 OUT OF THE 4 POSSIBLE OPTIONS HAVE *
1060*   BEEN ELIMINATED. IF THIS IS TRUE THE OPTIMAL DECISION*
1070*   HAS BEEN MADE AND IS STORED IN THE ARRAY 'OPDECL'      *
1080C
1090       IF(TYPRPR.EQ.0111) OPDECL(I)=LOCAT(1)
1100       IF(TYPRPR.EQ.1011) OPDECL(I)=LOCAT(2)
1110       IF(TYPRPR.EQ.1101) OPDECL(I)=LOCAT(3)
1120       IF(TYPRPR.EQ.1110) OPDECL(I)=LOCAT(4)
1130       TSECD(I)=0
1140       TSECB(I)=0
1150       TSECC(I)=0
1160C
1170*   DO LOOP 25 UPDATES THE NUMBER OF PIECES OF SE AT EACH *
1180*   LEVEL BASED ON THE REPAIR LEVEL DECISIONS THAT HAVE *
1190*   BEEN MADE AS WELL AS THE SE HOURS STILL AVAILABLE.   *
1200*   IT THEN GOES BACK AND CHECKS TO MAKE SURE THAT ONLY *
1210*   ONE ADDITIONAL PIECE OF SE IS SUFFICIENT.             *
1220C
1230       DO 25 I25=1,26
1240           PTR=SEARY(I,I25)
1250           IF(PTR.EQ.0) GO TO 25
1260           HRAVSE(PTR)=NSECI(PTR)*(OPHRS(PTR)-BSYHRS(PTR))
1270           ID=SECODE(PTR)/10
1280           IF(LFMOB(I).GT.0) GO TO 41
1290           IF(TYPRPR.EQ.0111.AND.ID.LT.5000) GO TO 42
1300 41 IF(LFMOB(I).GT.0) GO TO 48
1310           IF(TYPRPR.EQ.1101.AND.ID.GE.5000.AND.ID.LT.9000) GO TO 44
1320 48 IF(LFMOC(I).GT.0) GO TO 10
1330           IF(TYPRPR.EQ.1110.AND.ID.GE.9000) GO TO 46
1340           GO TO 25
1350 42 IF(HRAVSE(PTR).GE.FSEUHD(I)) GO TO 43
1360           REQMT(PTR)=REQMT(PTR)+1
1370           HRAVSE(PTR)=HRAVSE(PTR)+OPHRS(PTR)
1380           GO TO 42
1390 43 USEHRS(PTR)=USEHRS(PTR)+FSEUHD(I)
1400           FAC=FDB(PTR)
1410           IF(NSECI(PTR).GT.0) FAC=0.
1420           TSECST(PTR)=TSECST(PTR)+((REQMT(PTR)*CADB(PTR)+FAC+
1430 & ((FSEUHD(I)/OPHRS(PTR))*CODE(PTR)*PIUP))/FLOAT(M))
1440           HRAVSE(PTR)=HRAVSE(PTR)-FSEUHD(I)
1450           GO TO 25
1460 44 IF(HRAVSE(PTR).GE.FSEUHB(I)) GO TO 45
1470           REQMT(PTR)=REQMT(PTR)+1
1480           HRAVSE(PTR)=HRAVSE(PTR)+OPHRS(PTR)
1490           GO TO 44
1500 45 USEHRS(PTR)=USEHRS(PTR)+FSEUHB(I)
1510           FAC=FDB(PTP)
1520           IF(NSECI(PTR).GT.0) FAC=0.
1530           TSECST(PTP)=TSECST(PTP)+((REQMT(PTR)*CADB(PTR)+FAC+((FSEUHB(I)
1540 & /OPHRS(PTR))*CODE(PTR)*PIUP))
1550           HRAVSE(PTP)=HRAVSE(PTR)-FSEUHB(I)
1560           GO TO 25

```

```

1570 46 IF(HRAVSE(PTR).GE.FSEUHC(I)) GO TO 47
1580 REQMT(PTR)=2+AINT((FSEUHC(I)*OS)/OPHRS(PTR))+
1590 & AINT((FSEUHC(I)*(1-OS))/OPHRS(PTR))
1600 HRAVSE(PTR)=HRAVSE(PTR)+(REQMT(PTR)*OPHRS(PTR))
1610 47 USEHRS(PTR)=USEHRS(PTR)+FSEUHC(I)
1620 FAC=FDB(PTR)
1630 IF(NSECI(PTR).GT.0) FAC=0.
1640 TSECST(PTR)=TSECST(PTR)+((REQMT(PTR)*CADB(PTR)+FAC+
1650 & ((FSEUHC(I)/OPHRS(PTR))*CODB(PTR)*PIUP))/FLOAT('')
1660 HRAVSE(PTR)=HRAVSE(PTR)-FSEUHC(I)
1670 25 CONTINUE
1680C
1690* IF THE CURRENT ITEM IS AN LRU EM GO TO 10 *
1700* BECAUSE THE FOLLOWING STATEMENTS RELATE ONLY TO SRU'S. *
1710C
1720 GO TO 10
1730 15 CONTINUE
1740 J=I-LFMS
1750 IF(OPDECL(LRUPTR(J)).EQ.LOCAT(15)) GO TO 10
1760 IF(SOD(J).NE.1) GO TO 22
1770 SOD(J)=2
1780 INDD=1000
1790 22 IF(SOB(J).NE.1) GO TO 24
1800 SOB(J)=2
1810 INDE=10
1820 24 IF(SOC(J).NE.1) GO TO 26
1830 SOC(J)=2
1840 INDC=1
1850 26 IF(SOS(J).NE.1) GO TO 28
1860 SOS(J)=2
1870 INDS=100
1880 28 CONTINUE
1890 IF(CIRF.EQ.0) SOC(I)=3
1900C
1910* DO LOOP 20 CHECKS EACH POSITION OF THE SE/LRU/SRU *
1920* CROSS REFERENCE ARRAY (SEARY) FOR THE POINTER VALUES *
1930* OF THE SE NECESSARY TO REPAIR THE Ith ITEM. *
1940C
1950 DO 50 I50=1,26
1960 PTR=SEARY(I,I50)
1970 IF(PTR.EQ.0) GO TO 50
1980 IP=SECODE(PTR)/10
1990 IF(SOD(J).GT.0.OR.ID.GE.5000) GO TO 52
2000 BUY=AINT(1.0+(SSEUHD(J)/OPHRS(PTR)))
2010 TSECJ(J)=TSECJ(J)+(BUY*CADB(PTR)+FDB(PTR)+(SSEUHD(J)/
2020 & OPHRS(PTR))*CODB(PTR)*PIUP)
2030 52 IF(SOB(J).GT.0.OR.ID.LT.5000.OR.ID.GE.9000) GO TO 54
2040 BUY=AINT(1.0+(SSEUHB(J)/OPHRS(PTR)))
2050 TSECJ(J)=TSECJ(J)+(BUY*CADB(PTR)+FDB(PTR)+((SSEUHB(J)/
2060 & OPHRS(PTR))*CODB(PTR)*PIUP))
2070 54 IF(SOC(J).GT.0.OR.ID.LT.9000) GO TO 50
2080 BUY=2+AINT((SSEUHC(J)*OS)/OPHRS(PTR))+AINT((SSEUHC(J)*(1-OS))
2090 & /OPHRS(PTR))

```

```

2100      TSECC(J)=TSECC(J)+(PIY*CADR(PTR)+FDB(PTR)+(SSCUNC(I)/
2110      & OPHRS(PTR))*CODR(PTR)*PIY)
2120      50 CONTINUE
2130C
2140C      NOW THE TOTAL COST FOR EACH ITEM IS
2150*      DETERMINED (TDCST,TBCST,AND TCCST). FINALLY USING A      *
2160*      SERIES OF IF STATEMENTS A MARGINAL ANALYSIS IS      *
2170*      PERFORMED TO IDENTIFY IF ONE REPAIR LEVEL IS MORE      *
2180*      ECONOMICAL THAN ANOTHER. IF THIS IS TRUE A UNIQUE      *
2190*      INDICATOR IS SET (INDD,INDB,INDS,INDC).      *
2200C
2210      IF(TSECD(J).EQ.0) GO TO 58
2220      SED=TSECD(J)/M
2230      IF(TSECC(J).EQ.0) GO TO 58
2240      SEC=TSECC(J)/M
2250      55 TDCST(J)=DEPOSC(J)+SED
2260      IF(OPDECL(LRUPTR(J)).EQ.LOCAT(3)) TDCST(J)=TDCST(J)+
2270      & BRLSS(J)+BRLDS(J)
2280      TBCST(J)=BASESC(J)+TSECB(J)
2290      TCCST(J)=CIRFSC(J)+SEC
2300      IF(OPDECL(LRUPTR(J)).EQ.LOCAT(4)) TCCST(J)=TCCST(J)+BRLCS(J)
2310      IF(OPDECL(LRUPTR(J)).EQ.LOCAT(3)) SCRPS(C(J)=SCRPS(C(J)+BRLSS(J)
2320      IF(CIRF.EQ.0) TCCST(J)=JUNTO
2330      IF(SOD(J).GT.0.OR.SOS(J).GT.0) GO TO 60
2340      IF(SCRPS(C(J).LT.DEPOSC(J)) INDD=1000
2350      IF(TDCST(J).LT.SCRPS(C(J)) INDS=100
2360      60 IF(SOS(J).GT.0.OR.SOB(J).GT.0) GO TO 62
2370      IF(TBCST(J).LT.SCRPS(C(J)) INDS=100
2380      IF(SCRPS(C(J).LT.BASESC(J)) INDB=10
2390      62 IF(SOB(J).GT.0.OR.SOD(J).GT.0) GO TO 64
2400      IF(TBCST(J).LT.DEPOSC(J)) INDD=1000
2410      IF(TDCST(J).LT.BASESC(J)) INDB=10
2420      64 IF(SOC(J).GT.0.OR.SOD(J).GT.0) GO TO 67
2430      IF(TCCST(J).LT.DEPOSC(J)) INDD=1000
2440      IF(TDCST(J).LT.CIRFSC(J)) INDC=1
2450      67 IF(SOC(J).GT.0.OR.SOS(J).GT.0) GO TO 68
2460      IF(SCRPLC(J).LT.CIRFSC(J)) INDC=1
2470      IF(TCCST(J).LT.SCRPS(C(J)) INDS=100
2480      68 IF(SOC(J).GT.0.OR.SOB(J).GT.0) GO TO 69
2490      IF(TCCST(J).LT.BASESC(J)) INDB=10
2500      IF(TBCST(J).LT.CIRFSC(J)) INDC=1
2510      IF(OPDECL(LRUPTR(J)).EQ.LOCAT(3)) SCRPS(C(J)=SCRPS(C(J)
2520      & -BPLSS(J)
2530C
2540C      LINES BELOW CHECK THE VALUES OF THE INDICATORS      *
2550*      IF THE INDICATOR HAS BEEN GIVEN THE RIGHT VALUE, THE      *
2560*      NRLA FAILURE MODE EXCLUSION ARRAY IS UPDATED TO      *
2570*      ELIMINATE THAT REPAIR OPTION FROM FURTHER CONSIDER-      *
2580*      ATION FOR THE Ith ITEM.      *
2590C
2600      69 IF(SOD(J).EQ.2) GO TO 61
2610      IF(INDD.EQ.1000) SOD(J)=1
2620      61 IF(SOS(J).EQ.2) GO TO 63

```

```

2630      IF(INDS.EQ.100 ) SOS(J)=1
2640 63 IF(SOB(J).EQ.2) GO TO 65
2650      IF(INDE.EQ.10)   SOB(J)=1
2660 65 IF(SOC(J).GE.2) GO TO 66
2670      IF(INDC.EQ.1)    SOC(J)=1
2680 66 TYPRPR=INDD+INDS+INDE+INDC
2690C
2700*      BASED ON THE VALUE OF TYPRPR THE BELOW STATEMENTS      *
2710*      CHECK TO SEE IF 3 OUT OF THE 4 POSSIBLE OPTIONS HAVE *
2720*      BEEN ELIMINATED. IF THIS IS TRUE THE OPTIMAL DECISION*
2730*      HAS BEEN MADE AND IS STORED IN THE ARRAY "OPDECS"      *
2740C
2750      IF(TYPRPR.EQ.0111) OPDECS(J)=LOCAT(1)
2760      IF(TYPRPR.EQ.1011) OPDECS(J)=LOCAT(2)
2770      IF(TYPRPR.EQ.1101) OPDECS(J)=LOCAT(3)
2780      IF(TYPRPR.EQ.1110) OPDECS(J)=LOCAT(4)
2790      TSECD(J)=0
2800      TSECB(J)=0
2810      TSECC(J)=0
2820C
2830*      DO LOOP 75 UPDATES THE NUMBER OF PIECES OF SE AT EACH *
2840*      LEVEL BASED ON THE REPAIR LEVEL DECISIONS THAT HAVE *
2850*      BEEN MADE AS WELL AS THE SE HOURS STILL AVAILABLE. *
2860*      IT THEN GOES BACK AND CHECKS TO MAKE SURE THAT ONLY *
2870*      ONE ADDITIONAL PIECE OF SE IS SUFFICIENT. *
2880C
2890      DO 75 I75=1,26
2900      PTR=SEARY(I,I75)
2910      IF(PTR.EQ.0) GO TO 75
2920      HRAVSE(PTR)=NSECI(PTR)*(OPHRS(PTR)-BSYHRS(PTR))
2930      ID=SECODE(PTR)/10
2940      IF(SOD(J).GT.0) GO TO 91
2950      IF(TYPRPR.EQ.0111.AND.ID.LT.5000) GO TO 92
2960 91 IF(SOB(J).GT.0) GO TO 98
2970      IF(TYPRPR.EQ.1101.AND.ID.GE.5000.AND.ID.LT.9000) GO TO 94
2980 98 IF(SOC(J).GT.0) GO TO 10
2990      IF(TYPRPR.EQ.1110.AND.ID.GE.9000) GO TO 95
3000      GO TO 75
3010 92 IF(HRAVSE(PTR).GE.SSEUHD(J)) GO TO 93
3020      REQMT(PTR)=REQMT(PTR)+1
3030      HRAVSE(PTR)=HRAVSE(PTR)+OPHRS(PTR)
3040      GO TO 92
3050 93 USEHRS(PTR)=USEHRS(PTR)+SSEUHD(J)
3060      FAC=FDB(PTR)
3070      IF(NSECI(PTR).GT.0) FAC=0.
3080      TSECST(PTR)=TSECST(PTR)+((REQMT(PTR)*CAGR(PTR)+FAC+
3090      & ((SSEUHD(J)/OPHRS(PTR))*COB(PTR)*PIUP))/FLOAT(M))
3100      HRAVSE(PTR)=HRAVSE(PTR)-SSEUHD(J)
3110      GO TO 75
3120 94 IF(HRAVSE(PTR).GE.SSEUHB(J)) GO TO 95
3130      REQMT(PTR)=REQMT(PTR)+1
3140      HRAVSE(PTR)=HRAVSE(PTR)+OPHRS(PTR)
3150      GO TO 94
3160 95 USEHRS(PTR)=USEHRS(PTR)+SSEUHB(J)
3170      FAC=FD3(PTR)

```

```

3180     IF(NSECI(PTR).GT.0) FAC=0.
3190     TSECST(PTR)=TSECST(PTR)+(TEOMT(PTR)*CAD3(PTR)+FAC+((SSEUHC(J)
3200     & /OPHRS(PTR))*COD3(PTR)*PIUP))
3210     HRAVSE(PTR)=HRAVSE(PTR)-SSEUHC(J)
3220     GO TO 75
3230 96 IF(HRAVSE(PTR).GE.SSEUHC(J)) GO TO 97
3240     REQHT(PTR)=2+AINT((SSEUHC(J)*OS)/OPHRS(PTR))+
3250     & AINT((SSEUHC(J)*(1-OS))/OPHRS(PTR))
3260     HRAVSE(PTR)=HRAVSE(PTR)+(REQHT(PTR)*OPHRS(PTR))
3270 97 USEHRS(PTR)=USEHRS(PTR)+SSEUHC(J)
3280     FAC=FDB(PTR)
3290     IF(NSECI(PTR).GT.0) FAC=0.
3300     TSECST(PTR)=TSECST(PTR)+((REQHT(PTR)*CAD3(PTR)+FAC+
3310     & ((SSEUHC(J)/OPHRS(PTR))*COD3(PTR)*PIUP))/FLOAT(M))
3320     HRAVSE(PTR)=HRAVSE(PTR)-SSEUHC(J)
3330 75 CONTINUE
3340     INDD=0
3350     INDC=0
3360     INDB=0
3370     INDS=0
3380 10 CONTINUE
3390     RETURN
3400     END

```

*

Appendix E: MSETNT Subroutine

```

10      SUBROUTINE MSETNT(MSGL2)
20C
30C
40C      CODE FROM STATEMENT # 2000 THROUGH # 2210 UTILIZES THE ABOVE
50C      COMPUTED ITEM & SE COSTS TO PRODUCE AN RLA COST NETWORK.
60C
70C      AS A PRELIMINARY STEP DETERMINE NODE NUMBERS FOR KEY NODES IN
80C      THE NETWORK. THE VARIABLES ARE CODED WITH THE PREFIXES 'LD'
90C      FOR 'LAST DEPOT' AND 'LB' FOR 'LAST BASE'.
100C
110 2000 LDSE=NDSE+1
120      LDLRU=LDSE+LFMS
130      LDSRU=LDLRU+SFMS
140      LBLRU=LDSRU+LFMS
150      LBSPU=LBLRU+SFMS
160      LBSE=LBSRU+NBSE
170      LNODE=LBSE+1
180      IF(LNODE.LE.MAXNOD) GO TO 2005
190      WRITE(6,2002)
200 2002 FORMAT('0 NODE VECTORS ARE TOO SMALL -- STOP')
210      STOP
220C
230C
240C      SET # OF ARCS TO ZERO & # OF NODES TO 1
250 2005 NARCS=0
260      NNODES=1
270C
280      IF(NDSE.LT.1) GO TO 2015
290C      CREATE ARCS FROM SOURCE TO DEPOT SE -- DEPOT SE COST ARCS
300      DO 2010 I2010=1,NDSE
310      NARCS=NARCS+1
320      NNODES=NNODES+1
330      SRCE(NARCS)=1
340      DEST(NARCS)=NNODES
350      CAP(NARCS)=SECCOST(I2010)
360 2010 CONTINUE
370C
380C      CREATE ARCS FROM THE SOURCE TO THE DEPOT LRU FAILURE MODE
390C      NODES -- ITEM RELATED COSTS FOR LRU REPAIR AT DEPOT
400 2015 DO 2020 I2020=1,LFMS
410      NARCS=NARCS+1
420      NNODES=NNODES+1
430      SRCE(NARCS)=1
440      DEST(NARCS)=NNODES
450      CAP(NARCS)=DEPOLC(I2020)
460 2020 CONTINUE
470C
480C      CREATE ARCS FROM THE SOURCE TO THE DEPOT SRU NODES -- ITEM
490C      RELATED COSTS FOR SRU REPAIR AT DEPOT
500      IF(SFMS.LT.1) GO TO 2040

```

```

510      DO 2030 I2030=1,SEMS
520      NARCS=NARCS+1
530      NNODES=NNODES+1
540      SRCE(NARCS)=1
550      DEST(NARCS)=NNODES
560      CAP(NARCS)=DEPOSC(I2030)
570 2030 CONTINUE
580C
590C      CREATE ARCS FROM THE DEPOT SE NODES TO THE DEPOT LRU/SRU NODES
600 2040 IF(NDSE.LT.1) GO TO 2070
610C      FOR EACH DEPOT SE --
620      DO 2060 I2060=1,NDSE
630C      SEPF CONTAINS A POINTER TO THE SE CROSS REFERENCE (SEXREF)
640      IPTR=SEPF(I2060)
650      IF(IPTR.EQ.0) GO TO 2060
660C      SEXREF CONTAINS THE # OF AN LRU FAILURE MODE (ITEM > 0) OR
670C      THE NUMBER OF AN SRU (ITEM < 0)
680 2050 ITEM=SEXREF(IPTR)
690      IF(ITEM.LT.0) ITEM=LFMS-ITEM
700      NARCS=NARCS+1
710      SRCE(NARCS)=I2060+1
720C      ITEM IS A POINTER TO AN LRU OR SRU AND WHEN ADDED TO LDSE
730C      GIVES THE APPROPRIATE NODE NUMBER
740      DEST(NARCS)=LDSE+ITEM
750      CAP(NARCS)=JUMBO
760C
770C      NXTITM IS A POINTER TO THE NEXT ENTRY IN SEXREF WHICH IS AN
780C      LRU OR SRU REQUIRING THE CURRENT SE, I.E., SE # I2060.
790      NPTR=NXTITM(IPTR)
800      IF(NPTR.EQ.0) GO TO 2060
810      IPTR=NPTR
820      GO TO 2050
830 2060 CONTINUE
840C
850C      CREATE THE ARCS EMANATING FROM THE DEPOT LRU FAILURE MODE NODES
860 2070 DO 2080 I2080=1,LFMS
870C      CHECK FOR AN SRU
880      IPTR=SRUPTR(I2080)
890      IF(IPTR.EQ.0) GO TO 2075
900C      CREATE AN ARC TO THE DEPOT SRU NODE -- COSTS UNIQUE TO BASE
910C      REPAIR OF LRU & DEPOT REPAIR OF SRU
920      NARCS=NARCS+1
930      SRCE(NARCS)=LDSE+I2080
940      DEST(NARCS)=LDLRU+IPTR
950      CAP(NARCS)=BRLDS(IPTR)
960C      CREATE AN ARC TO THE BASE LRU NODES -- COST OF SCRAPPING THE LRU
970 2075 NARCS=NARCS+1
980      SRCE(NARCS)=LDSE+I2080
990      DEST(NARCS)=LDSRU+I2080
1000      CAP(NARCS)=SCRPLC(I2080)
1010 2080 CONTINUE
1020C

```

```

1030C      CREATE ARCS FROM THE DEPOT SRU NODES TO THE BASE SRU NODES --
1040C      COST OF SCRAPPING THE SRU
1050      IF(SFMS.LT.1) GO TO 2100
1060      DO 2090 I2090=1,SFMS
1070      NARCS=NARCS+1
1080      SRCE(NARCS)=LDBLRU+I2090
1090      DEST(NARCS)=LDBLRU+I2090
1100      CAP(NARCS)=SCRPSG(I2090)
1110 2090 CONTINUE
1120C
1130C      CREATE ARCS EMANATING FROM THE BASE LRU NODES
1140 2100 DO 2120 I2120=1,LFMS
1150C      CHECK FOR AN SRU
1160      IPTR=SRUPTR(I2120)
1170      IF(IPTR.EQ.0) GO TO 2110
1180C
1190C      CREATE THE ARC FROM THE BASE LRU NODE TO THE BASE SRU NODE --
1200C      COSTS UNIQUE TO BASE REPAIR OF THE LRU AND SCRAPPING (OR DEPOT
1210C      REPAIR) OF THE SRU
1220      NARCS=NARCS+1
1230      SRCE(NARCS)=LDSRU+I2120
1240      DEST(NARCS)=LDBLRU+IPTR
1250      CAP(NARCS)=BRLSS(IPTR)
1260C      CREATE AN ARC FROM THE BASE LRU NODE TO THE LAST NODE (SINK) --
1270C      ITEM RELATED COST OF BASE REPAIR OF THE LRU
1280 2110 NARCS=NARCS+1
1290      SRCE(NARCS)=LDSRU+I2120
1300      DEST(NARCS)=LNODE
1310      CAP(NARCS)=BASELC(I2120)
1320 2120 CONTINUE
1330C
1340C      CREATE ARCS EMANATING FROM THE BASE SRU NODES
1350      IF(SFMS.LT.1) GO TO 2140
1360      DO 2130 I2130=1,SFMS
1370C      CREATE AN ARC FROM THE BASE SRU NODE TO THE BASE LRU NODE --
1380C      THIS ARC PREVENTS A DECISION TO DO BASE LEVEL SRU REPAIR UNLESS
1390C      THE LRU IS ALSO BASE REPAIRED
1400      NARCS=NARCS+1
1410      SRCE(NARCS)=LDBLRU+I2130
1420      DEST(NARCS)=LDSRU+LPUPTTR(I2130)
1430      CAP(NARCS)=JUMBO
1440C
1450C      CREATE AN ARC FROM THE BASE SRU NODE TO THE LAST NODE (SINK) --
1460C      ITEM RELATED COST OF BASE REPAIR FOR THE SRU
1470      NARCS=NARCS+1
1480      SRCE(NARCS)=LDBLRU+I2130
1490      DEST(NARCS)=LNODE
1500      CAP(NARCS)=BASESC(I2130)
1510 2130 CONTINUE
1520C
1530C      CREATE ARCS TO AND FROM THE BASE SE NODES
1540 2140 IF(NBSE.LT.1) GO TO 2180
1550C      FOR EACH BASE SE --

```



```

1560      NDSEPI=NDSE+1
1570      NBSEPO=NDSE+NBSE
1580      DO 2170 I2170=NDSEPI,NBSEPO
1590C      J2170 IS A POINTER TO THE BASE SE
1600      J2170=I2170-NDSE
1610C      SEPF CONTAINS A POINTER TO THE SE CROSS REFERENCE (SEXREF)
1620C      ( SEE COMMENTS FOR STATEMENTS 2040 TO 2060)
1630      IPTR=SEPF(I2170)
1640      IF(IPTR.EQ.0) GO TO 2160
1650 2150 ITEM=SEXREF(IPTR)
1660      IF(ITEM.LT.0) ITEM=LFMS-ITEM
1670      NARCS=NARCS+1
1680      SRCE(NARCS)=LDSRU+ITEM
1690      DEST(NARCS)=LBSRU+J2170
1700      CAP(NARCS)=JUMBO
1710C
1720      NPTR=NXTITM(IPTR)
1730      IF(NPTR.EQ.0) GO TO 2160
1740      IPTR=NPTR
1750      GO TO 2150
1760C
1770C      CREATE AN ARC FROM THE BASE SE TO THE LAST NODE (SINK) --
1780C      BASE SE COST
1790 2160 NARCS=NARCS+1
1800      SRCE(NARCS)=LBSRU+J2170
1810      DEST(NARCS)=LNODE
1820      CAP(NARCS)=SEPCOST(I2170)
1830 2170 CONTINUE
1840 2180 NNODES=LNODE
1850C
1860C      EXPLICITLY SORT THE SRCE-DEST-CAP TRIPLES INTO ASCENDING ORDER
1870C      BY SRCE (FLOW IS USED BY SORT TO STORE ORDERING POINTERS)
1880      CALL SORT(SRCE,DEST,CAP,FLOW,NARCS,MAXARC,1)
1890C      SAVE POINTERS FOR FORWARD SCAN IN FWDSP
1900      DO 2190 I2190=1,NARCS
1910C      UTILIZE THE VALUES IN SRCE IN REVERSE ORDER. (IF NODE '1' IS A
1920C      SOURCE FOR 'N' ARCS THEN FWDSP(1) WILL SUCCESSIVELY GET THE
1930C      VALUES: N, N-1, N-2, . . . , 2, 1. SIMILARLY, FWDSP(K) WILL
1940C      HAVE THE VALUE 'J' WHERE J CORRESPONDS TO THE FIRST OCCURENCE
1950C      OF NODE # K IN SRCE.)
1960      J2190=NARCS+1-I2190
1970      K2190=SRCE(J2190)
1980      FWDSP(K2190)=J2190
1990 2190 CONTINUE
2000      FWDSP(LNODE)=NARCS+1
2010      LNODM1=LNODE-1
2020      DO 2195 I2195=1,LNODM1
2030      J2195=LNODE-I2195
2040      IF(FWDSP(J2195).EQ.0) FWDSP(J2195)=FWDSP(J2195+1)
2050 2195 CONTINUE
2060C
2070C      USE SORT TO IMPLICITLY SEQUENCE THE DEST ENTRIES INTO ASCENDING
2080C      ORDER. THE ORDER WILL BE SPECIFIED BY THE ENTRIES IN BKPTR.

```

```

2090C      THUS, THE ENTRIES IN BACKSP ARE POINTERS TO SRCE-DEST-CAP
2100C      TRIPLES BASED ON THE DEST VALUES.
2110      CALL SORT(DEST,0,0,BKPTR,NARCS,MAXARC,0)
2120      BACKSP(1)=0
2130      DO 2200 I2200=1,NARCS
2140      J2200=BKPTR(I2200)
2150      K2200=DEST(J2200)
2160C      BACKSP(K) IS THE POSITION IN BKPTR WHICH HOLDS THE LAST POINTER
2170C      TO A SRCE-DEST-CAP TRIPLE HAVING NODE # K AS THE DEST. THE FIRST
2180C      POINTER TO A TRIPLE HAVING NODE # K AS THE DEST IS BKPTR(K-1)+1.
2190      BACKSP(K2200)=I2200
2200 2200 CONTINUE
2210      DO 2222 I2222=2,LNODE
2220      IF(BACKSP(I2222).EQ.0) BACKSP(I2222)=BACKSP(I2222-1)
2230 2222 CONTINUE
2240C
2250C IF(MSGL2.LT.1) GO TO 2205
2260C PRINT 2202,'SRCE ',(SRCE(I),I=1,NARCS)
2270C 2202 FORMAT(' ',A6,(' ',20I3))
2280C PRINT 2202,'DEST ',(DEST(I),I=1,NARCS)
2290C PRINT 2202,'CAP ',(CAP(I),I=1,NARCS)
2300C PRINT 2202,'FLOW ',(FLOW(I),I=1,NARCS)
2310C PRINT 2202,'BKPTR ',(BKPTR(I),I=1,NARCS)
2320C PRINT 2202,'FWDSP ',(FWDSP(I),I=1,LNODE)
2330C PRINT 2202,'BACKSP',(BACKSP(I),I=1,LNODE)
2340 2205 DO 2400 I2400=1,NARCS
2350      J=SRCE(I2400)
2360      K=DEST(I2400)
2370      IF(J.NE.1) GO TO 2230
2380      IF(K.GT.LDSE) GO TO 2210
2390      SEARCP(K-1)=I2400
2400      GO TO 2300
2410 2210 IF(K.GT.LDLRU) GO TO 2220
2420      LDARC(K-LDSE)=I2400
2430      GO TO 2300
2440 2220 SDARC(K-LDLRU)=I2400
2450      GO TO 2300
2460 2230 IF(J.LE.LDSE) GO TO 2300
2470      IF(J.GT.LDLRU) GO TO 2250
2480      IF(K.LE.LDSRU) GO TO 2240
2490      LSARC(J-LDSE)=I2400
2500      GO TO 2300
2510 2240 SBDARC(K-LDLRU)=I2400
2520      GO TO 2300
2530 2250 IF(J.GT.LDSRU) GO TO 2260
2540      SSARC(J-LDLRU)=I2400
2550      GO TO 2300
2560 2260 IF(J.GT.LBLRU) GO TO 2280
2570      IF(K.NE.LNODE) GO TO 2270
2580      LBARC(J-LDSRU)=I2400
2590      GO TO 2300
2600 2270 IF(K.GT.LBSRU) GO TO 2300
2610      SBSARC(K-LBLRU)=I2400

```

```

2620      GO TO 2300
2630 2280 IF(J.GT.LBSRU) GO TO 2290
2640      IF(K.NE.LNODE) GO TO 2300
2650      SBARC(J-LBLRU)=I2400
2660      GO TO 2300
2670 2290 SEARCHP(J+NDSE-LBSRU)=I2400
2680 2300 FLOW(I2400)=0
2690 2400 CONTINUE
2700C
2710C      THIS ROUTINE USED ONLY IF MARLA IS CALLED IN MAIN
2720C
2730      IF(MAR.EQ.0) GO TO 3000
2740      DO 2450 I2450=1,MAXMOD
2750          FWDSP(I2450)=0
2760          BACKSP(I2450)=0
2770 2450 CONTINUE
2780      DO 2460 I2460=1,MAXARC
2790          BKPTR(I2460)=0
2800 2460 CONTINUE
2810C
2820      DO 2181 I2181=1,LFMS
2830      IF(OPDECL(I2181).EQ.LOCAT(15)) GO TO 2181
2840          IPTR=LDARC(I2181)
2850          SRCE(IPTR)=LNODE+10
2860          IPTR=LBARC(I2181)
2870          SRCE(IPTR)=LNODE+10
2880          IPTR=LSARC(I2181)
2890          SRCE(IPTR)=LNODE+10
2900      IF(OPDECL(I2181).NE.LOCAT(3)) GO TO 2191
2910          CPTR=SRUPTR(I2181)
2920          TC1=BRLDS(CPTR)
2930          TC2=BRLSS(CPTR)
2940          W=SDARC(CPTR)
2950          CAP(W)=CAP(W)+TC1+TC2
2960          X=SSARC(CPTR)
2970          CAP(X)=CAP(X)+TC2
2980          IPTR=SBSARC(CPTR)
2990          SRCE(IPTR)=LNODE+10
3000          IPTR=SDARC(CPTR)
3010          SRCE(IPTR)=LNODE+10
3020 2191 DO 2182 I2182=1,NARCS
3030          IF(DEST(I2182).EQ.(LDSE+I2181)) SRCE(I2182)=LNODE+10
3040          IF(SRCE(I2182).EQ.(LDSE+I2181)) SRCE(I2182)=LNODE+10
3050          IF(SRCE(I2182).EQ.(LDSRU+I2181)) SRCE(I2182)=LNODE+10
3060          IF(DEST(I2182).EQ.(LDSRU+I2181)) SRCE(I2182)=LNODE+10
3070 2182 CONTINUE
3080 2181 CONTINUE
3090      DO 2184 I2184=1,SFMS
3100      IF(OPDECS(I2184).EQ.LOCAT(15)) GO TO 2184
3110          IPTR=SDARC(I2184)
3120          SRCE(IPTR)=LNODE+10
3130          IPTR=SEARC(I2184)
3140          SRCE(IPTR)=LNODE+10

```

```

3150      IPTR=SSARC(I2184)
3160      SRCE(IPTR)=LNODE+10
3170      DO 2185 I2185=1,NARCS
3180      IF(DEST(I2185).EQ.(LDLRU+I2184)) SRCE(I2185)=LNODE+10
3190      IF(SRCE(I2185).EQ.(LBLRU+I2184)) SRCE(I2185)=LNODE+10
3200      IF(DEST(I2185).EQ.(LBLRU+I2184)) SRCE(I2185)=LNODE+10
3210 2185  CONTINUE
3220 2184  CONTINUE
3230C     EXPLICITLY SORT THE SRCE-DEST-CAP TRIPLES INTO ASCENDING ORDER
3240C     BY SRCE (FLOW IS USED BY SORT TO STORE ORDERING POINTERS)
3250      CALL SORT(SRCE,DEST,CAP,FLOW,NARCS,MAXARC,1)
3260      ITER=0
3270      DO 2487 I2487=1,NARCS
3280      IF(SRCE(I2487).LT.LNODE) ITER=ITER+1
3290 2487  CONTINUE
3300      NARCS=ITER
3310C     SAVE POINTERS FOR FORWARD SCAN IN FWDSP
3320      DO 2490 I2490=1,NARCS
3330C     UTILIZE THE VALUES IN SRCE IN REVERSE ORDER. (IF NODE '1' IS A
3340C     SOURCE FOR 'N' ARCS THEN FWDSP(1) WILL SUCCESSIVELY GET THE
3350C     VALUES: N, N-1, N-2, . . . , 2, 1. SIMILARLY, FWDSP(K) WILL
3360C     HAVE THE VALUE 'J' WHERE J CORRESPONDS TO THE FIRST OCCURENCE
3370C     OF NODE # K IN SRCE.)
3380      J2490=NARCS+1-I2490
3390      K2490=SRCE(J2490)
3400      FWDSP(K2490)=J2490
3410 2490  CONTINUE
3420      FWDSP(LNODE)=NARCS+1
3430      LNODM1=LNODE-1
3440      DO 2495 I2495=1,LNODM1
3450      J2495=LNODE-I2495
3460      IF(FWDSP(J2495).EQ.0) FWDSP(J2495)=FWDSP(J2495+1)
3470 2495  CONTINUE
3480C
3490C     USE SORT TO IMPLICITLY SEQUENCE THE DEST ENTRIES INTO ASCENDING
3500C     ORDER. THE ORDER WILL BE SPECIFIED BY THE ENTRIES IN BKPTR.
3510C     THUS, THE ENTRIES IN BACKSP ARE POINTERS TO SRCE-DEST-CAP
3520C     TRIPLES BASED ON THE DEST VALUES.
3530      CALL SORT(DEST,0,0,BKPTR,NARCS,MAXARC,0)
3540      BACKSP(1)=0
3550      DO 2500 I2500=1,NARCS
3560      J2500=BKPTR(I2500)
3570      K2500=DEST(J2500)
3580C     BACKSP(K) IS THE POSITION IN BKPTR WHICH HOLDS THE LAST POINTER
3590C     TO A SRCE-DEST-CAP TRIPLE HAVING NODE # K AS THE DEST. THE FIRST
3600C     POINTER TO A TRIPLE HAVING NODE # K AS THE DEST IS BKPTR(K-1)+1.
3610      BACKSP(K2500)=I2500
3620 2500  CONTINUE
3630      DO 2525 I2525=2,LNODE
3640      IF(BACKSP(I2525).EQ.0) BACKSP(I2525)=BACKSP(I2525-1)
3650 2525  CONTINUE
3660C
3670      DO 2600 I2600=1,NARCS

```

```

3680      J=SPCE(I2600)
3690      K=DEST(I2600)
3700      IF(J.NE.1) GO TO 2530
3710      IF(K.GT.LDSE) GO TO 2510
3720      SEARCP(K-1)=I2600
3730      GO TO 2595
3740 2510 IF(K.GT.LDLRU) GO TO 2520
3750      LDARC(K-LDSE)=I2600
3760      GO TO 2595
3770 2520 SDARC(K-LDLRU)=I2600
3780      GO TO 2595
3790 2530 IF(J.LE.LDSE) GO TO 2595
3800      IF(J.GT.LDLRU) GO TO 2550
3810      IF(K.LE.LDSRU) GO TO 2540
3820      LSARC(J-LDSE)=I2600
3830      GO TO 2595
3840 2540 SBDARC(K-LDLRU)=I2600
3850      GO TO 2595
3860 2550 IF(J.GT.LDSRU) GO TO 2560
3870      SSARC(J-LDLRU)=I2600
3880      GO TO 2595
3890 2560 IF(J.GT.LBLRU) GO TO 2580
3900      IF(K.NE.LNODE) GO TO 2570
3910      LBARC(J-LDSRU)=I2600
3920      GO TO 2595
3930 2570 IF(K.GT.LBSRU) GO TO 2595
3940      SBSARC(K-LBLRU)=I2600
3950      GO TO 2595
3960 2580 IF(J.GT.LBSRU) GO TO 2590
3970      IF(K.NE.LNODE) GO TO 2595
3980      SBARC(J-LBLRU)=I2600
3990      GO TO 2595
4000 2590 SEARCP(J+NDSE-LBSRU)=I2600
4010 2595 FLOW(I2600)=0
4020 2600 CONTINUE
4030 3000 CONTINUE
4040C IF(MSGL2.LT.2) GO TO 2440
4050C DO 2410 I=1,LFMS
4060C PRINT,'LPTRS ',LDARC(I),LSARC(I),LBARC(I)
4070C 2410 CONTINUE
4080C DO 2420 I=1,SFMS
4090C PRINT,'SPTRS ',SDARC(I),SSARC(I),SBARC(I),SBDARC(I),SBSARC(I)
4100C 2420 CONTINUE
4110C DO 2430 I=1,NUMSER
4120C PRINT,'SEPTRS ',SEARCP(I)
4130C 2430 CONTINUE
4140C 2440 CONTINUE
4150      RETURN
4160      END

```

Appendix F: NMXFLO Subroutine

```

10      SUBROUTINE NMXFLO(MSGL2)
20C
30C
40C      TEMPORARY DATA INPUT AREAS
50      INTEGER MIRROR,NSTART,IAPREV,NAPATH(300)
60C
70      IF (ICIRF.EQ.2) GOTO 3012
80      DO 3005 I3005=1,LFMS
90      IF (OPDECL(I3005).NE.LOCAT(15)) GOTO 3005
100     IF(LFMOD(I3005).EQ.0) GO TO 3001
110     J3005=LDARC(I3005)
120     CAP(J3005)=JUMBO
130 3001 IF(LFMOS(I3005).EQ.0) GO TO 3002
140     J3005=LSARC(I3005)
150     CAP(J3005)=JUMBO
160 3002 IF(LFMOB(I3005).EQ.0) GO TO 3005
170     J3005=LBARC(I3005)
180     CAP(J3005)=JUMBO
190 3005 CONTINUE
200C
210     IF(SFMS.LT.1) GO TO 3012
220     DO 3010 I3010=1,SFMS
230     IF (OPDECS(I3010).NE.LOCAT(15)) GOTO 3010
240     IF(SOD(I3010).EQ.0) GO TO 3006
250     J3010=SDARC(I3010)
260     CAP(J3010)=JUMBO
270 3006 IF(SOS(I3010).EQ.0) GO TO 3007
280     J3010=SSARC(I3010)
290     CAP(J3010)=JUMBO
300 3007 IF(SOB(I3010).EQ.0) GO TO 3010
310     J3010=SBARC(I3010)
320     CAP(J3010)=JUMBO
330 3010 CONTINUE
340 3012 CONTINUE
350     CALL PTIME(X)
360C
370C      DEPTH FIRST MAX FLOW ALGOTITHM
380C
390C      DATA USED IS CONTAINED IN THE VECTORS SRCE, DEST ,CAP, FLOW
400C      AND BKPTR WHICH HAVE AN ENTRY FOR EACH ARC; AND IN VECTORS
410C      NPATH, DLTAFL, STATE, FWDSP, AND BACKSP WHICH HAVE AN ENTRY FOR
420C      EACH NODE. STATE IS A '1' FOR AN UNLABELED NODE AND A '2' FOR A
430C      LABELED NODE.
440C
450C      LABEL THE SOURCE
460C
470      DLTAFL(1) = JUMBO
480      MIRROR = 0
490      NPATH(1) = -1
500 5015 STATE(1) = 2

```

```

510C
520C LABEL ALL OTHER NODES AS UNLABELED '1'
530C
540 DO 5020 I5020 = 2,LNODE
550 STATE(I5020) = 1
560 5020 CONTINUE
570C
580C FIND AN ARC FROM NODE 1 WITH CAPACITY REMAINING
590C
600 J5050 = FWDSP(1)
610 K5050 = FWDSP(2) - 1
620 DO 5060 I5060 = J5050,K5050
630 NODE = DEST(I5060)
640 IF (FLOW(I5060).LT.CAP(I5060).AND.STATE(NODE).EQ.1) GOTO 5065
650 GOTO 5060
660C
670C MARK FORWARD OPATH AND CALC MAX FLOW ON ARC I5060
680C
690 5065 NPATH(NODE) = 1
700 STATE(NODE) = 2
710 NAPATH(NODE) = I5060
720 JDELTA = CAP(I5060) - FLOW(I5060)
730 DLTAFL(NODE) = MIN0(JDELTA,DLTAFL(1))
740C
750C CONTINUE FORWARD DEPTH FIRST FLOW AUGMENTATION
760C
770 5070 NNODE = NODE
780 J5080 = FWDSP(NNODE)
790 K5080 = FWDSP(NNODE + 1) - 1
800 DO 5090 I5090 = J5080,K5080
810 NODE = DEST(I5090)
820 IF (FLOW(I5090).LT.CAP(I5090).AND.STATE(NODE).EQ.1) GOTO 5085
830 GOTO 5090
840 5085 NPATH(NODE) = NNODE
850 STATE(NODE) = 2
860 NAPATH(NODE) = I5090
870 JDELTA = CAP(I5090) - FLOW(I5090)
880 DLTAFL(NODE) = MIN0(JDELTA,DLTAFL(NNODE))
890C
900C IF NODE EQUALS LAST NODE INCREMENT FLOW ON PATH
910C
920 IF (NODE.EQ.LNODE) GOTO 5150
930 GOTO 5070
940 5090 CONTINUE
950C
960C FIRST TIME THROUGH SKIP MIRROR ARC SECTION
970C
980 IF (MIRROR.EQ.0) GOTO 5140
990C
1000C MIRROR ARC AUGMENTING PATH SECTION
1010C
1020 5100 J5100 = BACKSP(NNODE - 1) + 1
1030 J5110 = BACKSP(NNODE)

```

```

1040      DO 5130 I5130 = J5100,J5110
1050      K5100 = BKPTR(I5130)
1060      NODE = SRCE(K5100)
1070C
1080C      CHECK FOR SOURCE NODE OR PATH JUST TRAVELED
1090C
1100      IF (NODE.EQ.1.OR.NODE.EQ.NPATH(NNODE)) GOTO 5130
1110      IF (FLOW(K5100).GT.0.AND.STATE(NODE).EQ.1) GOTO 5120
1120      GOTO 5130
1130C
1140C      MARK REVERSE PATH AND CALC MAX FLOW ON ARC K5100
1150C
1160 5120  NPATH(NODE) = 0 - NNODE
1170      STATE(NODE) = 2
1180      NAPATH(NODE) = K5100
1190      DLTAFL(NODE) = MINO(FLOW(K5100),DLTAFL(NNODE))
1200      GOTO 5070
1210 5130  CONTINUE
1220C
1230C      IF UNABLE TO AUGMENT PATH TO SINK FROM THIS NODE
1240C      LABEL IT '2' AND BACKUP TO PREVIOUS NODE ON PATH
1250C
1260 5140  NODE = IABS(NPATH(NNODE))
1270      IF (NODE.EQ.1) GOTO 5060
1280      GOTO 5070
1290C
1300C      FLOW AUGMENTATION ON PATH FROM SOURCE TO SINK
1310C
1320 5150  LAST = LNODE
1330      STATE(LAST) = 1
1340      NSTART = 0
1350      INC = DLTAFL(LNODE)
1360 5160  IPREV = NPATH(LAST)
1370      IAPREV = NAPATH(LAST)
1380      IF (IPREV.GT.0) GOTO 5170
1390      FLOW(IAPREV) = FLOW(IAPREV) - INC
1400      GOTO 5180
1410 5170  FLOW(IAPREV) = FLOW(IAPREV) + INC
1420 5180  LAST = IABS(IPREV)
1430      IF (LAST.EQ.1) GOTO 5200
1440      DLTAFL(LAST) = DLTAFL(LAST) - INC
1450C
1460C      IDENTIFY AND MARK THE LAST NODE IN THE PATH WITH EXCESS
1470C      FLOW REMAINING FOR AUGMENTATION TO THE SINK
1480C
1490      IF (FLOW(IAPREV).LT.CAP(IAPREV).AND.DLTAFL(LAST).EQ.0)
1500      & STATE(LAST) = 1
1510      IF (NSTART.EQ.1.OR.DLTAFL(LAST).EQ.0) GOTO 5190
1520      NSTART = 1
1530      NODE = LAST
1540 5190  GOTO 5160

```



```

1550 5200 IF (NSTART.EQ.1) GOTO 5070
1560 5060 CONTINUE
1570C
1580C AFTER SECOND TIME THROUGH SOLUTION COMPLETE
1590C
1600 IF (MIRROR.EQ.2) GOTO 3300
1610C
1620C SET SWITCH TO ALLOW FOR FLOW AUGMENTATION ON MIRROR ARCS
1630C AND START OVER
1640C
1650 MIRROR = MIRROR + 1
1660 GOTO 5015
1670C
1680C
1690C ***** NETWORK SOLUTION COMPLETE *****
1700C COMPUTE MAX FLOW
1710 3300 J3310=1
1720 K3310=FWSP(2)-1
1730 ITFLOW=0
1740 DO 3310 I3310=J3310,K3310
1750 ITFLOW=ITFLOW+FLOW(I3310)
1760 3310 CONTINUE
1770 FLOW(MAXARC+1)=ITFLOW
1780C IF(MSGL2.LT.1) GO TO 3400
1790C PRINT,' TOTAL FLOW ',ITFLOW
1800C PRINT 2220,'NODE #',(I,I=1,LNODE)
1810C PRINT 2220,'SRCE ',(SRCE(I),I=1,NARCS)
1820C PRINT 2220,'DEST ',(DEST(I),I=1,NARCS)
1830C PRINT 2220,'NPATH ',(NPATH(I),I=1,LNODE)
1840C PRINT 2220,'DLTAFL',(DLTAFL(I),I=1,LNODE)
1850C PRINT 2220,'STATE ',(STATE(I),I=1,LNODE)
1860C PRINT 2220,'FLOW ',(FLOW(I),I=1,NARCS)
1870C PRINT 2220,'CAP ',(CAP(I),I=1,NARCS)
1880C 3400 CONTINUE
1890 LCSEC=0
1900 NDSEP1=NDSE+1
1910 IF(NDSE.EQ.0) GO TO 3330
1920 DO 3320 I3320=2,NDSEP1
1930 IF(STATE(I3320).NE.1) GO TO 3320
1940 SEPTR=SEARCP(I3320-1)
1950 LCSEC=LCSEC+CAP(SEPTR)
1960 3320 CONTINUE
1970 3330 IF(NDSE.EQ.NUMSER) GO TO 3350
1980 DO 3340 I3340=NDSEP1,NUMSER
1990 J3340=LBSRU+I3340-NDSE
2000 IF(STATE(J3340).EQ.1) GO TO 3340
2010 SEPTR=SEARCP(I3340)
2020 LCSEC=LCSEC+CAP(SEPTR)
2030 3340 CONTINUE
2040 3350 FLOW(MAXARC+2)=LCSEC
2050C J=0

```

```

2060C DO 3360 I=1,NARCS
2070C IF(DEST(I).EQ.LNODE) J=J+FLOW(I)
2080C 3360 CONTINUE
2090C IF(J.NE.ITFLOW) PRINT,'NOT CONSISTENT FLOW ',ITFLOW,J
2100     Y = 0
2110     CALL PTIME(Y)
2120     Z=Y-X
2130     TT=TT+Z
2140     WRITE(6,400) X*3600,Y*3600,Z*3600,TT*3600
2150 400 FORMAT(' TIME=',4F10.5)
2160     RETURN
2170     END

```

*

Appendix G: OUTPUT Subroutine

```

10      SUBROUTINE OUTPUT
20C
30C
40C
50C      OUTPUT RESULTS FOR DEPOT SE
60 3400 IPAGE=IPAGE+1
70      WRITE(6,3401) (SYSNAM(ISUB),ISUB=1,3),DATE,IPAGE
80 3401 FORMAT('1',2X,3A4,86X,A8,5X,'PAGE',I4)
90      WRITE(6,3402)
100 3402 FORMAT(' ',24X,'NETWORK RLA RESULTS')
110      WRITE(6,3403)
120 3403 FORMAT(' ',90X,'WHOLESALE CHANGE FACTORS')
130      WRITE(6,3404) WCFUC,WCFE,WCFSE
140 3404 FORMAT(' ',96X,'COST =',F6.2/97X,'MTBF =',F6.2/99X,'SE =',F6.2)
150      WRITE(6,3410)
160 3410 FORMAT(' ',22X,'SUPPORT EQUIPMENT REQUIPEMENTS')
170      WRITE(6,3412)
180 3412 FORMAT('0',4X,'DEPOT',28X,'TOTAL SE')
190      WRITE(6,3415)
200 3415 FORMAT(' ',5X,'SE CODE',5X,'NAME',10X,'REQMT',5X,'LC S/BASE',3X,
210      & 'USE HRS',7X,'USE %')
220      TOTSED=0.
230      IF(NDSE.LT.1) GO TO 3431
240      DO 3430 I3430=1,NDSE
250      IF(STATE(I3430+1).GT.1) GO TO 3430
260      WRITE(6,3420) SECODE(I3430),(XSE(I3430,ISUB),ISUB=1,3),
270      & REQMT(I3430),TSECST(I3430),USEHRS(I3430),SEUR(I3430)
280 3420 FORMAT(' ',2X,I5,5X,3A4,3X,I3,1X,F12.0,F12.0,2PF12.1)
290      TOTSED=TOTSED+TSECST(I3430)
300 3430 CONTINUE
310 3431 CONTINUE
320      WRITE(6,3435) TOTSED
330 3435 FORMAT('0',2X,'TOTAL',24X,F12.0)
340C
350C      OUTPUT RESULTS FOR BASE SUPPORT EQUIPMENT
360      IF(NDSE.EQ.NUMSER) GO TO 3451
370      WRITE(6,3440)
380 3440 FORMAT('0',4X,'BASE',29X,'TOTAL SE')
390      WRITE(6,3415)
400      TOTSEB=0.
410      NDSFPI=NDSE+1
420      NBSEPO=NDSE+NBSE
430      DO 3450 I3450=NDSFPI,NBSEPO
440      J3450=LBSPU+I3450-NDSE
450      IF(STATE(J3450).EQ.1) GO TO 3450
460 3445 WRITE(6,3420) SECODE(I3450),(XSE(I3450,ISUB),ISUB=1,3),
470      & REQMT(I3450),SECCST(I3450),USEHRS(I3450),SEUR(I3450)
480      TOTSEB=TOTSEB+TSECST(I3450)
490 3450 CONTINUE
500      WRITE(6,3435) TOTSEB

```

```

510C   OUTPUT RESULTS FOR CIRF SUPPORT EQUIPMENT
520 3442 IF (NBSEP0.EQ.NUMSER) GOTO 3451
530    WRITE(6,3800)
540 3800 FORMAT('0',4X,'CIRF',29X,'TOTAL SF')
550    WRITE (6,3415)
560    IF (ICIRF.EQ.1) WRITE (6,3810)
570 3810 FORMAT('  NO CIRF SE CALCULATED THIS PASS')
580    IF (ICIRF.EQ.1) GOTO 3451
590    TOTSEC=0
600    NBSEP1=NBSEP0+1
610    DO 3820 I3820=NBSEP1,NUMSER
620    J3820=I3820-NBSEP0
630    IF (STATE(J3820).GT.1) GOTO 3820
640    WRITE (6,3420) SECODE(I3820),(XSE(I3820,ISUB),ISUB=1,3),
650    & REQMT(I3820),TSECST(I3820),USEHRS(I3820),SEPR(I3820)
660    TOTSEC=TOTSEC+TSECST(I3820)
670 3820 CONTINUE
680    WRITE (6,3435) TOTSEC
690C
700C   OUTPUT RESULTS FOR LRU'S & SRU'S
710 3451 IPAGE=IPAGE+1
720    WRITE(6,3401) (SYSNAM(ISUB),ISUB=1,3),DATE,IPAGE
730    WRITE(6,3452) CHARS(2),CHARS(2)
740 3452 FORMAT(' ',40X,'LRU & SRU REPAIR LEVEL DECISIONS',2X,2A4)
750    WRITE(6,3453)
760 3453 FORMAT('0',29X,'-- LRU (LC $/BASE) --',7X,'-- SRU ',
770    & '(LC $/BASE) --',7X,'LRU/SRU  LRU',7X,'INPUT  MTB ',
780    & 'LC REP'/' ',4X,'LRU/SRU NAME IDENT NO  DEPOT  SCRAP',
790    & 'BASE', ' CIRF', 'DEPOT  SCRAP  BASE', ' CIRF ',
800    & 'COST  FAIL%',7X,'MTBF  REPAIR DEM/BASE')
810    TOTLD=0.
820    TOTLS=0.
830    TOTLC=0.
840    TOTLB=0.
850    TOTSD=0.
860    TOTSS=0.
870    TOTSB=0.
880    LINES=3
890    WUC(1)=LOCAT(15)
900    WUC(2)=LOCAT(15)
910    NDSEP2=NDSE+2
920    NSE=NDSEP2
930    ICIRF=1
940    LLRU=LDLRU
950    DO 3600 I3600=NSE,LLRU
960    ITEM=I3600-(NSE-1)
970    IF(LFMWUC(ITEM,1).EQ.WUC(1).AND.LFMWUC(ITEM,2).EQ.WUC(2))
980    & GO TO 3457
990    DO 3454 I3454=1,NLRU
1000    IF(LFMWUC(ITEM,1).EQ.LMWUC(I3454,1).AND.
1010    & LFMWUC(ITEM,2).EQ.LMWUC(I3454,2)) LPTR=I3454
1020 3454 CONTINUE
1030    MTBCT=MTBF(LPTR)/(UF(LPTR)*(1.-RIP(LPTR)))

```

```

1040      TOCTGM=PGMR*(PA(LPTR)/MTBCT
1050      TLCD=TOCTGM*PIUP*12.
1060 3457 IF(LINES.LT.54) GO TO 3458
1070      IPAGE=IPAGE+1
1080      WRITE(6,3401) (SYSNAM(ISUB),ISUB=1,3),DATE,IPAGE
1090      WRITE(6,3452) CHARS(5),CHARS(6)
1100      WRITE(6,3453)
1110      LINES=4
1120 3458 TLCDF=TLCD*FAILP(ITEM)
1130      FMTBCT=MTBCT/FAILP(ITEM)
1140      NODEL=I360G
1150      NODER=NODEL+LFMS+SFMS
1160      IF (OPDECL(ITEM).EQ.LOCAT(4)) GOTO 3483
1170      IF (OPDECL(ITEM).EQ.LOCAT(1)) GOTO 3460
1180      IF (OPDECL(ITEM).EQ.LOCAT(2)) GOTO 3470
1190      IF (OPDECL(ITEM).EQ.LOCAT(3)) GOTO 3480
1200C      CHECK FOR LRU DEPOT REPAIR
1210      IF(STATE(NODEL).EQ.1.AND.STATE(NODER).EQ.1) GO TO 3460
1220C      CHECK FOR LRU SCRAPPED
1230      IF(STATE(NODEL).NE.1.AND.STATE(NODER).EQ.1) GO TO 3470
1240C      CHECK FOR LRU BASE REPAIRED
1250      IF(STATE(NODEL).NE.1.AND.STATE(NODER).NE.1) GO TO 3480
1260      WRITE(6,3459)
1270 3459 FORMAT(' TROUBLE ON LRU LABELS')
1280      STOP
1290C
1300 3460 COSTL=DEPOLC(ITEM)
1310      TOTLD=TOTLD+COSTL
1320      LEVEL=1
1330      OPDECL(ITEM)=LOCAT(1)
1340      GO TO 3490
1350C
1360 3470 COSTL=SCRPLC(ITEM)
1370      TOTLS=TOTLS+COSTL
1380      LEVEL=2
1390      OPDECL(ITEM)=LOCAT(2)
1400      GO TO 3490
1410C
1420 3480 COSTL=BASELC(ITEM)
1430      TOTLB=TOTLB+COSTL
1440      LEVEL=3
1450      OPDECL(ITEM)=LOCAT(3)
1460      GOTO 3490
1470 3483 COSTL=CIRFLC(ITEM)
1480      TOTLC=TOTLC+COSTL
1490      LEVEL=4
1500      OPDECL(ITEM)=LOCAT(4)
1510 3490 FPCT=FAILP(ITEM)*100.
1520      LOD=CHARS(2)
1530      LOS=CHARS(2)
1540      LOB=CHARS(2)
1550      LOC=CHARS(2)
1560      IF(LFMOD(ITEM).EQ.2) LOD=CHARS(14)

```

```

1570     IF(LFMOB(ITEM).EQ.2)  LOS=CHARS(14)
1580     IF(LFMOC(ITEM).EQ.2)  LOB=CHARS(14)
1590     IF (LFMOC(ITEM).EQ.2)  LOC=CHARS(14)
1600     IF(SRUPTR(ITEM).NE.0)  GO TO 3520
1610     GO TO (3493,3500,3510,3580),LEVEL
1620 3493 IF(WUC(1).NE.LFMWUC(ITEM,1).OR.WUC(2).NE.LFMWUC(ITEM,2))
1630     &    GO TO 3496
1640 3494 WRITE(6,3495)  COSTL,LOS,LOB,LOC,FPCT,FMTBCT,TLCDF
1650 3495 FORMAT(' ',27X,F8.0,3X,A4,4X,A4,2X,A4,38X,F5.1,13X,F8.0,2X,F8.1)
1660     LINES=LINES+1
1670     GO TO 3600
1680 3496 IF(FRSTFM(LPTR).EQ.LASTFM(LPTR))  GO TO 3498
1690     WRITE(6,3497)  LPTR,LRUNAM(LPTR,1),LRUNAM(LPTR,2),LRUNAM(LPTR,3),
1700     &    LWUC(LPTR,1),LWUC(LPTR,2),UCL(LPTR),MTBF(LPTR),MTBCT,TLCD
1710 3497 FORMAT('0',I2,2X,3A4,2X,A4,A3,59X,F8.0,10X,F8.0,2X,F8.0,
1720     &    2X,F8.1)
1730     LINES=LINES+2
1740     WUC(1)=LFMWUC(ITEM,1)
1750     WUC(2)=LFMWUC(ITEM,2)
1760     GO TO 3494
1770 3498 WRITE(6,3499)  LPTR,LRUNAM(LPTR,1),LRUNAM(LPTR,2),LRUNAM(LPTR,3),
1780     &    LWUC(LPTR,1),LWUC(LPTR,2),COSTL,LOS,LOB,LOC,UCL(LPTR),
1790     &    FPCT,MTBF(LPTR),FMTBCT,TLCDF
1800 3499 FORMAT('0',I2,2X,3A4,2X,A4,A3,2X,F8.0,3X,A4,4X,A4,2X,A4,28X,F8.0,
1810     &    2X,F5.1,3X,F8.0,2X,F8.0,2X,F8.1)
1820     LINES=LINES+2
1830     WUC(1)=LFMWUC(ITEM,1)
1840     WUC(2)=LFMWUC(ITEM,2)
1850     GO TO 3600
1860C
1870 3500 IF(WUC(1).NE.LFMWUC(ITEM,1).OR.WUC(2).NE.LFMWUC(ITEM,2))
1880     &    GO TO 3503
1890 3501 WRITE(6,3502)  LOD,COSTL,LOB,LOC,FPCT,FMTBCT,TLCDF
1900 3502 FORMAT(' ',30X,A4,1X,F8.0,3X,A4,2X,A4,38X,F5.1,13X,F8.0,2X,F8.1)
1910     LINES=LINES+1
1920     GO TO 3600
1930 3503 IF(FRSTFM(LPTR).EQ.LASTFM(LPTR))  GO TO 3504
1940     WRITE(6,3497)  LPTR,LRUNAM(LPTR,1),LRUNAM(LPTR,2),LRUNAM(LPTR,3),
1950     &    LWUC(LPTR,1),LWUC(LPTR,2),UCL(LPTR),MTBF(LPTR),MTBCT,TLCD
1960     LINES=LINES+2
1970     WUC(1)=LFMWUC(ITEM,1)
1980     WUC(2)=LFMWUC(ITEM,2)
1990     GO TO 3501
2000 3504 WRITE(6,3505)  LPTR,LRUNAM(LPTR,1),LRUNAM(LPTR,2),LRUNAM(LPTR,3),
2010     &    LWUC(LPTR,1),LWUC(LPTR,2),LOD,COSTL,LOB,LOC,
2020     &    UCL(LPTR),FPCT,MTBF(LPTR),FMTBCT,TLCDF
2030 3505 FORMAT('0',I2,2X,3A4,2X,A4,A3,5X,A4,1X,F8.0,3X,A4,2X,A4,28X,
2040     &    F8.0,2X,F5.1,3X,F8.0,2X,F8.0,2X,F8.1)
2050     LINES=LINES+2
2060     WUC(1)=LFMWUC(ITEM,1)
2070     WUC(2)=LFMWUC(ITEM,2)
2080     GO TO 3600
2090C

```

```

2100 3510 IF(WUC(1).NE.LFMWUC(ITEM,1).OR.WUC(2).NE.LFMWUC(ITEM,2))
2110      &      GO TO 3513
2120 3511 WRITE(6,3512)  LOD,LOS,COSTL,LOC,FPCT,FMTBCT,TLCDF
2130 3512 FORMAT(' ',30X,A4,4X,A4,1X,F8.0,1X,A4,38X,F5.1,13X,F8.0,2X,F8.1)
2140      LINES=LINES+1
2150      GO TO 3600
2160 3513 IF(FRSTFM(LPTR).EQ.LASTFM(LPTR)) GO TO 3514
2170      WRITE(6,3497) LPTR,LRUNAM(LPTR,1),LRUNAM(LPTR,2),LRUNAM(LPTR,3),
2180      &      LWUC(LPTR,1),LWUC(LPTR,2),UCL(LPTR),
2190      &      MTBF(LPTR),MTBCT,TLCD
2200      LINES=LINES+2
2210      WUC(1)=LFMWUC(ITEM,1)
2220      WUC(2)=LFMWUC(ITEM,2)
2230      GO TO 3511
2240 3514 WRITE(6,3515) LPTR,LRUNAM(LPTR,1),LRUNAM(LPTR,2),LRUNAM(LPTR,3),
2250      &      LWUC(LPTR,1),LWUC(LPTR,2),LOD,LOS,COSTL,LOC,
2260      &      UCL(LPTR),FPCT,MTBF(LPTR),FMTBCT,TLCDF
2270 3515 FORMAT('0',I2,2X,3A4,2X,A4,A3,5X,A4,4X,A4,1X,F8.0,1X,A4,28X,F8.0,2X,
2280      &      F5.1,3X,F8.0,2X,F8.0,2X,F8.1)
2290      LINES=LINES+2
2300      WUC(1)=LFMWUC(ITEM,1)
2310      WUC(2)=LFMWUC(ITEM,2)
2320      GO TO 3600
2330C
2340 3580 IF(WUC(1).NE.LFMWUC(ITEM,1).OR.WUC(2).NE.LFMWUC(ITEM,2))
2350      &      GO TO 3583
2360 3581 WRITE(6,3582)  LOD,LOS,LOB,COSTL,FPCT,FMTBCT,TLCDF
2370 3582 FORMAT(' ',30X,A4,3X,A4,3X,A4,1X,F7.0,38X,F5.1,13X,F8.0,2X,F8.1)
2380      LINES=LINES+1
2390      GO TO 3600
2400 3583 IF(FRSTFM(LPTR).EQ.LASTFM(LPTR)) GO TO 3584
2410      WRITE(6,3497) LPTR,LRUNAM(LPTR,1),LRUNAM(LPTR,2),LRUNAM(LPTR,3),
2420      &      LWUC(LPTR,1),LWUC(LPTR,2),UCL(LPTR),
2430      &      MTBF(LPTR),MTBCT,TLCD
2440      LINES=LINES+2
2450      WUC(1)=LFMWUC(ITEM,1)
2460      WUC(2)=LFMWUC(ITEM,2)
2470      GO TO 3581
2480 3584 WRITE(6,3585) LPTR,LRUNAM(LPTR,1),LRUNAM(LPTR,2),LRUNAM(LPTR,3),
2490      &      LWUC(LPTR,1),LWUC(LPTR,2),LOD,LOS,LOB,COSTL,
2500      &      UCL(LPTR),FPCT,MTBF(LPTR),FMTBCT,TLCDF
2510 3585 FORMAT('0',I2,2X,3A4,2X,A4,A3,5X,A4,3X,A4,3X,A4,1X,F7.0,29X,
2520      &      F8.0,2X,F5.1,3X,F8.0,2X,F8.0,2X,F8.1)
2530      LINES=LINES+2
2540      WUC(1)=LFMWUC(ITEM,1)
2550      WUC(2)=LFMWUC(ITEM,2)
2560      GO TO 3600
2570C
2580 3520 IF(WUC(1).EQ.LFMWUC(ITEM,1).AND.WUC(2).EQ.LFMWUC(ITEM,2))
2590      &      GO TO 3525
2600      WRITE(6,3497) LPTR,LRUNAM(LPTR,1),LRUNAM(LPTR,2),LRUNAM(LPTR,3),
2610      &      LWUC(LPTR,1),LWUC(LPTR,2),UCL(LPTR),
2620      &      MTBF(LPTR),MTBCT,TLCD

```

```

2630      LINES=LINES+2
2640      WUC(1)=LFMWUC(ITEM,1)
2650      WUC(2)=LFMWUC(ITEM,2)
2660C
2670 3525 NODEL=LDLRU+SRUPTR(ITEM)
2680      NODER=LBLRU+SRUPTR(ITEM)
2690      ITEM=SRUPTR(ITEM)
2700      ISOD=CHARS(2)
2710      ISOS=CHARS(2)
2720      ISOB=CHARS(2)
2730      ISOC=CHARS(2)
2740      IF(SOD(ITEM).EQ.2) ISOD=CHARS(14)
2750      IF(SOS(ITEM).EQ.2) ISOS=CHARS(14)
2760      IF(SOB(ITEM).EQ.2) ISOB=CHARS(14)
2770      IF(SOC(ITEM).EQ.2) ISOC=CHARS(14)
2780      IF (OPDECS(ITEM).EQ.LOCAT(4)) GOTO 3680
2790      IF (ICIRF.EQ.2.AND.STATE(NODEL).EQ.1) GOTO 3680
2800      IF (OPDECS(ITEM).EQ.LOCAT(1)) GOTO 3530
2810      IF (OPDECS(ITEM).EQ.LOCAT(2)) GOTO 3550
2820      IF (OPDECS(ITEM).EQ.LOCAT(3)) GOTO 3570
2830C      CHECK FOR SRU DEPOT REPAIR
2840      IF(STATE(NODEL).EQ.1.AND.STATE(NODER).EQ.1) GO TO 3530
2850C      CHECK FOR SRU SCRAPPED
2860      IF(STATE(NODEL).NE.1.AND.STATE(NODER).EQ.1) GO TO 3550
2870C      CHECK FOR SRU BASE REPAIRED
2880      IF(STATE(NODEL).NE.1.AND.STATE(NODER).NE.1) GO TO 3570
2890      WRITE(6,3526)
2900 3526 FORMAT(' TROUBLE ON SRU LABELS')
2910      STOP
2920C      SRU IS DEPOT REPAIRED
2930 3530 COST=DEPOSC(ITEM)
2940C      IF LRU IS BASE REPAIRED ADD EXTRA COSTS
2950      IF(LEVEL.EQ.3) COST=COST+BRLDS(ITEM)+BRLSS(ITEM)
2960C
2970      TOTSD=TOTSD+COST
2980      JPDECS(ITEM)=LOCAT(1)
2990      GO TO (3532,3534,3536,3538),LEVEL
3000 3532 WRITE(6,3533) SRUNAM(ITEM,1),SRUNAM(ITEM,2),SWUC(ITEM,1),
3010      & SWUC(ITEM,2),COSTL,LOS,LOB,LOC,COST,ISOS,ISOB,ISOC,UCS(ITEM),
3020      & FPCT,FMTBCT,TLCDF
3030 3533 FORMAT(' ',8X,2A4,2X,A4,A3,2X,F8.0,3X,A4,4X,A4,1X,A4,F8.0,
3040      & 3X,A4,4X,A4,2X,A4,F8.0,2X,F5.1,13X,F8.0,2X,F3.1)
3050      GO TO 3590
3060 3534 WRITE(6,3535) SRUNAM(ITEM,1),SRUNAM(ITEM,2),SWUC(ITEM,1),
3070      & SWUC(ITEM,2),LOD,COSTL,LOB,LOC,COST,ISOS,ISOB,ISOC,UCS(ITEM),
3080      & FPCT,FMTBCT,TLCDF
3090 3535 FORMAT(' ',3X,2A4,2X,A4,A3,5X,A4,1X,F8.0,3X,A4,1X,A4,F8.0,
3100      & 3X,A4,4X,A4,2X,A4,F8.0,2X,F5.1,
3110      & 13X,F8.0,2X,F8.1/40X,'CASE 7 ERROR IN ABOVE DECISION')
3120      GO TO 3590
3130 3536 WRITE(6,3537) SRUNAM(ITEM,1),SRUNAM(ITEM,2),SWUC(ITEM,1),
3140      & SWUC(ITEM,2),LOD,LOS,COSTL,LOC,COST,ISOS,ISOB,ISOC,UCS(ITEM),
3150      & FPCT,FMTBCT,TLCDF

```



```

3160 3537 FORMAT(' ',8X,2A4,2X,A4,A3,5X,A4,4X,A4,1X,F7.0,1X,A4,F8.0,
3170      &      3X,A4,4X,A4,2X,A4,F8.0,2X,F5.1,
3180      &      13X,F8.0,2X,F8.1)
3190      GO TO 3590
3200 3538 WRITE(6,3539) SRUNAM(ITEM,1),SRUNAM(ITEM,2),SWUC(ITEM,1),
3210      &      SWUC(ITEM,2),LOD,LOS,LOB,COSTL,COST,ISOS,ISOB,ISOC,UCS(ITEM),
3220      &      FPCT,FMTBCT,TLCDF
3230 3539 FORMAT(' ',8X,2A4,2X,A4,A3,5X,A4,4X,A4,2X,A4,F7.0,F8.0,
3240      &      3X,A4,4X,A4,2X,A4,F8.0,2X,F5.1,13X,F8.0,2X,F8.1)
3250      GO TO 3590
3260C
3270C      SRU IS SCRAPPED
3280 3550 COST=SCRPSC(ITEM)
3290C      IF LRU IS BASE REPAIRED ADD EXTRA COST
3300      IF(LEVEL.EQ.3) COST=COST+BRLSS(ITEM)
3310      TOTSS=TOTSS+COST
3320      OPDECS(ITEM)=LOCAT(2)
3330      GO TO (3552,3554,3556,3558),LEVEL
3340 3552 WRITE(6,3553) SRUNAM(ITEM,1),SRUNAM(ITEM,2),SWUC(ITEM,1),
3350      &      SWUC(ITEM,2),COSTL,LOS,LOB,LOC,ISOD,COST,ISOB,ISOC,UCS(ITEM),
3360      &      FPCT,FMTBCT,TLCDF
3370 3553 FORMAT(' ',8X,2A4,2X,A4,A3,2X,F8.0,3X,A4,4X,A4,2X,A4,2X,A4,1X,
3380      &      F8.0,3X,A4,2X,A4,F8.0,2X,F5.1,
3390      &      13X,F8.0,2X,F8.1)
3400      GO TO 3590
3410 3554 WRITE(6,3555) SRUNAM(ITEM,1),SRUNAM(ITEM,2),SWUC(ITEM,1),
3420      &      SWUC(ITEM,2),LOD,COSTL,LOB,LOC,ISOD,COST,ISOB,ISOC,UCS(ITEM),
3430      &      FPCT,FMTBCT,TLCDF
3440 3555 FORMAT(' ',8X,2A4,2X,A4,A3,5X,A4,1X,F8.0,3X,A4,2X,A4,2X,A4,1X,
3450      &      F8.0,3X,A4,2X,A4,F8.0,2X,F5.1,
3460      &      13X,F8.0,2X,F8.1)
3470      GO TO 3590
3480 3556 WRITE(6,3557) SRUNAM(ITEM,1),SRUNAM(ITEM,2),SWUC(ITEM,1),
3490      &      SWUC(ITEM,2),LOD,LOS,COSTL,LOC,ISOD,COST,ISOB,ISOC,UCS(ITEM),
3500      &      FPCT,FMTBCT,TLCDF
3510 3557 FORMAT(' ',8X,2A4,2X,A4,A3,5X,A4,4X,A4,1X,F8.0,1X,A4,2X,A4,1X,
3520      &      F8.0,3X,A4,2X,A4,F8.0,3X,F4.0,
3530      &      13X,F8.0,2X,F8.1)
3540      GO TO 3590
3550C
3560 3558 WRITE(6,3559) SRUNAM(ITEM,1),SRUNAM(ITEM,2),SWUC(ITEM,1),
3570      &      SWUC(ITEM,2),LOD,LOS,LOB,COSTL,ISOD,COST,ISOB,ISOC,UCS(ITEM),
3580      &      FPCT,FMTBCT,TLCDF
3590 3559 FORMAT(' ',8X,2A4,2X,A4,A3,5X,A4,4X,A4,2X,A4,1X,F7.0,2X,A4,1X,
3600      &      F8.0,3X,A4,2X,A4,F8.0,3X,F4.0,
3610      &      13X,F8.0,2X,F8.1)
3620      GO TO 3590
3630C
3640C      SRU IS BASE REPAIRED
3650 3570 COST=BASESC(ITEM)
3660      TOTSB=TOTSB+COST
3670      OPDECS(ITEM)=LOCAT(3)
3680      GO TO (3572,3574,3576,3578),LEVEL

```

```

3690 3572 WRITE(6,3573) SRUNAM(ITEM,1),SRUNAM(ITEM,2),SWUC(ITEM,1),
3700 & SWUC(ITEM,2),COSTL,LOS,LOB,LOC,ISOD,ISOS,COST,ISOC,UCS(ITEM),
3710 & FPCT,FMTBCT,TLCDF
3720 3573 FORMAT(' ',8X,2A4,2X,A4,A3,2X,F8.0,3X,A4,4X,A4,2X,A4,2X,A4,4X,
3730 & A4,1X,F8.0,'ERROR',F8.0,2X,F5.1,
3740 & 13X,F8.0,2X,F8.1)
3750 GO TO 3590
3760 3574 WRITE(6,3575) SRUNAM(ITEM,1),SRUNAM(ITEM,2),SWUC(ITEM,1),
3770 & SWUC(ITEM,2),LOD,COSTL,LOB,LOC,ISOD,ISOS,COST,ISOC,UCS(ITEM),
3780 & FPCT,FMTBCT,TLCDF
3790 3575 FORMAT(' ',8X,2A4,2X,A4,A3,5X,A4,1X,F8.0,3X,A4,2X,A4,2X,A4,4X,
3800 & A4,1X,F8.0,'ERROR',F8.0,2X,F5.1,
3810 & 13X,F8.0,2X,F8.1)
3820 GO TO 3590
3830 3576 WRITE(6,3577) SRUNAM(ITEM,1),SRUNAM(ITEM,2),SWUC(ITEM,1),
3840 & SWUC(ITEM,2),LOD,LOS,COSTL,LOC,ISOD,ISOS,COST,ISOC,UCS(ITEM),
3850 & FPCT,FMTBCT,TLCDF
3860 3577 FORMAT(' ',8X,2A4,2X,A4,A3,5X,A4,4X,A4,1X,F8.0,1X,A4,2X,A4,4X,
3870 & A4,1X,F8.0,1X,A4,F8.0,2X,F5.1,
3880 & 13X,F8.0,2X,F8.1)
3890 GOTO 3590
3900 3578 WRITE(6,3579) SRUNAM(ITEM,1),SRUNAM(ITEM,2),SWUC(ITEM,1),
3910 & SWUC(ITEM,2),LOD,LOS,LOB,COSTL,ISOD,ISOS,COST,ISOC,UCS(ITEM),
3920 & FPCT,FMTBCT,TLCDF
3930 3579 FORMAT(' ',8X,2A4,2X,A4,A3,5X,A4,4X,A4,2X,A4,F7.0,2X,A4,4X,
3940 & A4,1X,F8.0,'ERROR',F8.0,2X,F5.1,
3950 & 13X,F8.0,2X,F8.1)
3960 GO TO 3590
3970C
3980C SRU IS CIRF REPAIRED
3990C
4000 3680 COST=CIRFSC(ITEM)
4010 TOTSC=TOTSC+COST
4020 OPDECS(ITEM)=LOCAT(4)
4030 GO TO (3682,3684,3686,3688),LEVEL
4040 3682 WRITE(6,3683) SRUNAM(ITEM,1),SRUNAM(ITEM,2),SWUC(ITEM,1),
4050 & SWUC(ITEM,2),COSTL,LOS,LOB,LOC,ISOD,ISOS,ISOB,COST,UCS(ITEM),
4060 & FPCT,FMTBCT,TLCDF
4070 3683 FORMAT(' ',8X,2A4,2X,A4,A3,2X,F8.0,3X,A4,4X,A4,2X,A4,2X,A4,4X,
4080 & A4,1X,'ERROR',F8.0,F8.0,2X,F5.1,
4090 & 13X,F8.0,2X,F8.1)
4100 GO TO 3590
4110 3684 WRITE(6,3685) SRUNAM(ITEM,1),SRUNAM(ITEM,2),SWUC(ITEM,1),
4120 & SWUC(ITEM,2),LOD,COSTL,LOB,LOC,ISOD,ISOS,ISOB,COST,UCS(ITEM),
4130 & FPCT,FMTBCT,TLCDF
4140 3685 FORMAT(' ',8X,2A4,2X,A4,A3,5X,A4,1X,F8.0,3X,A4,2X,A4,2X,A4,4X,
4150 & A4,1X,'ERROR',F8.0,F8.0,2X,F5.1,
4160 & 13X,F8.0,2X,F8.1)
4170 GO TO 3590
4180 3686 WRITE(6,3687) SRUNAM(ITEM,1),SRUNAM(ITEM,2),SWUC(ITEM,1),
4190 & SWUC(ITEM,2),LOD,LOS,COSTL,LOC,ISOD,ISOS,ISOB,COST,UCS(ITEM),
4200 & FPCT,FMTBCT,TLCDF
4210 3687 FORMAT(' ',8X,2A4,2X,A4,A3,5X,A4,4X,A4,1X,F8.0,1X,A4,2X,A4,4X,

```

```

4220      &      A4,2X,A4,F7.0,1X,F8.0,2X,F5.1,
4230      &      13X,F8.0,2X,F8.1)
4240      GOTO 3590
4250 3688 WRITE(6,3689) SRUNAM(ITEM,1),SRUNAM(ITEM,2),SWUC(ITEM,1),
4260      &      SWUC(ITEM,2),LOD,LOS,LOB,COSTL,ISOD,ISOS,ISOB,COST,UCS(ITEM),
4270      &      FPCT,FMTBCT,TLCDF
4280 3689 FORMAT(' ',8X,2A4,2X,A4,A3,5X,A4,4X,A4,2X,A4,F7.0,2X,A4,4X,
4290      &      A4,1X,'ERROR',F8.0,F8.0,2X,F5.1,
4300      &      13X,F8.0,2X,F8.1)
4310C
4320 3590 LINES=LINES+1
4330 3600 CONTINUE
4340C
4350C      WRITE COST TOTALS
4360      WRITE(6,3604)
4370 3604 FORMAT(' ',39X,'----- LC $/BASE -----')
4380      WRITE(6,3605)
4390 3605 FORMAT('0',35X,7X,'DEPOT',7X,'SCRAP',8X,'BASE',7X,'CIRF',14X,
4400      &      'WHOLESALE CHANGE FACTORS')
4410      WRITE(6,3610) TOTLD,TOTLS,TOTLB,TOTLC,WCFUC
4420 3610 FORMAT(' TOTAL LRU COSTS',20X,4F12.0,19X,'COST =' ,F6.2)
4430      WRITE(6,3620) TOTSD,TOTSS,TOTSB,TOTSC,WCFB
4440 3620 FORMAT(' TOTAL SRU COSTS',20X,4F12.0,19X,'MTRF =' ,F6.2)
4450      WRITE(6,3630) TOTSED,TOTSEB,TOTSEC,WCFSE
4460 3630 FORMAT(' ', 'TOTAL SE COSTS',21X,F12.0,12X,F12.0,12X,F12.0,
4470      &      9X,'SE =' ,F6.2)
4480      DEV=DEV/M
4490      WRITE(6,3635) DEV
4500 3635 FORMAT(' ', 'SE DEVELOPMENT COST',24X,F12.0)
4510      GTOT=TOTLD+TOTLS+TOTLB+TOTSD+TOTSS+TOTSB+TOTSED+TOTSEB+DEV
4520      &      +TOTLC+TOTSEC+TOTSC
4530      WRITE(6,3640) GTOT
4540 3640 FORMAT('0',5X,'TOTAL COST OF REPAIR LEVEL DECISIONS =' ,
4550      &      F12.0/20('*'))
4560      RETURN
4570      END

```

Appendix H: SECMP Subroutine

```

10      SUBROUTINE SECMP(IND,MSCL1)
20C
30C
40      IF(IND.EQ.0) GO TO 806
50      IPAGE=IPAGE+1
60      800 WRITE(6,801) (SYSNAM(ISUB),ISUB=1,3),DATE,IPAGE
70      801 FORMAT('1',2X,3A4,86X,A8,5X,'PAGE',I4)
80      WRITE(6,802) CHARS(2),CHARS(2)
90      802 FORMAT(' ',43X,'COMPUTED SE COSTS',2X,2A4)
100     WRITE(6,803) PIUP
110     803 FORMAT('0',4X,'SE',9X,'SE',9X,'*** NO. ***   *** SE USE ***'/
120     & ' ',3X,'CODE',7X,'NAME',8X,'CURR NEW',8X,'HRS/MO ',4X,
130     & ' %USE',15X,F4.0,' YR',14X,'TOTAL SE')
140     WRITE(6,804)
150     804 FORMAT(' ',26X,'INV REQD CURR NEW',12X,'ACQ. $',4X,
160     & ' OPER. $',5X,'FAC. $',2X,'LC $/BASE')
170     WRITE(6,805)
180     805 FORMAT(' DEPOT')
190     LINES=7
200     IF(NDSE.GT.0) GO TO 806
210     WRITE(6,8050)
220     8050 FORMAT(' BASE')
230     LINES=LINES+1
240     IF (NBSE.GT.0) GOTO 806
250     WRITE(6,8051)
260     8051 FORMAT(' CIRF')
270C
280     806 IF(NUMSER.LT.1) RETURN
290     DO 900 I900=1,NUMSER
300     SAVCST=SECCOST(I900)
310C     COMPUTE TOTAL USAGE HOURS FOR THE SE
320     ITYPE=SECCODE(I900)/10000
330     JTYPE=(SECCODE(I900)/1000)-(ITYPE*10)
340     TMHRS=USEHRS(I900)
350     TMQRT=REQMT(I900)
360     USEHRS(I900)=0.
370     NEXT=SEPF(I900)
380     IF(NEXT.EQ.0) GO TO 815
390     LAST=SEPL(I900)
400     807 ITMPTR=SEXREF(NEXT)
410     IF(ITMPTR.GT.0.AND.OPDECL(ITMPTR).NE.LOCAT(15)) GOTO 811
420     IF(ITMPTR.LT.0.AND.OPDECS(0-ITMPTR).NE.LOCAT(15)) GOTO 811
430     IF(ITYPE.GT.4.AND.ITYPE.LT.9) GO TO 809
440     IF(ITYPE.EQ.9) GOTO 8090
450     IF(ITMPTR.LT.0) GO TO 808
460     USEHRS(I900)=USEHRS(I900)+FSEUHD(ITMPTR)
470     GO TO 811
480     808 ITMPTR=0-ITMPTR
490     USEHRS(I900)=USEHRS(I900)+SSEUHD(ITMPTR)
500     GO TO 811
510     809 IF(ITMPTR.LT.0) GO TO 810

```

```

520     USEHRS(I900)=USEHRS(I900)+FSEUHB(ITMPTR)
530     GO TO 811
540 8090 IF(ITMPTR.LT.0) GOTO 8095
550     USEHRS(I900)=USEHRS(I900)+FSEUHC(ITMPTR)
560     GOTO 811
570 3095 ITMPTR=0-ITMPTR
580     USEHRS(I900)=USEHRS(I900)+SSEUHC(ITMPTR)
590     GOTO 811
600 810 ITMPTR=0-ITMPTR
610     USEHRS(I900)=USEHRS(I900)+SSEUHB(ITMPTR)
620 811 IF(NEXT.EQ.LAST) GO TO 314
630     NEXT=NXTITM(NEXT)
640     IF(NEXT.GT.0) GO TO 807
650 812 WRITE(6,813) I900,SEPF(I900),SEPL(I900),NEXT,SEXREF(NEXT),
660     & NXTITM(NEXT)
670 813 FORMAT('OTROUBLE IN SEXREF',6I6)
680     STOP
690 814 IF(NXTITM(NEXT).EQ.0) GO TO 815
700     WRITE(6,8140)
710 8140 FORMAT(' NEXT = 0')
720     GO TO 812
730C     COMPUTE # OF UNITS OF THE SE REQUIRED
740 815 IF(ITYPE.EQ.1.OR.ITYPE.EQ.5.OR.JTYPE.EQ.1) GO TO 817
750C     FOR PECULIAR SE
760     IF (JTYPE.GT.1) GOTO 816
770     IF (USEHRS(I900).GT.HRAVSE(I900)) GOTO 818
780     REQMT(I900)=0
790     GOTO 819
800 816 TOTHR=TMHRS+USEHRS(I900)
810     REQMT(I900)= AINT(2.+((TOTHR*OS)/OPHRS(I900)))+
820     & AINT((TOTHR*(1.-OS))/OPHRS(I900))
830     IF(TMRQT.GT.0) REQMT(I900)=REQMT(I900)-TMRQT
840     GOTO 819
850 818 REQMT(I900)= AINT(1+((USEHRS(I900)-HRAVSE(I900))/OPHRS(I900)))
860 819 SEUR(I900)=(USEHRS(I900)+TMHRS)/((REQMT(I900)+TMRQT)*OPHRS(I900))
870     IF(ITYPE.NE.4.AND.ITYPE.NE.8.AND.JTYPE.NE.4) GO TO 840
880     REQMT(I900)=1
890     TMRQT=0
900     SEUR(I900)=0.
910     GO TO 840
920C     FOR COMMON SE
930 817 REQMT(I900)=0.
940     IF (JTYPE.EQ.1) GOTO 821
950     AVHRS=(OPHRS(I900)-BSYHRS(I900))*NSECI(I900)-TMHRS
960 820 IF(AVHRS.GT.USEHRS(I900)) GO TO 830
970     REQMT(I900)=REQMT(I900)+1.
980     AVHRS=AVHRS+OPHRS(I900)
990     GO TO 820
1000 821 TOTHR=TMHRS+USEHRS(I900)+BSYHRS(I900)
1010     REQMT(I900)=2.+AINT((TOTHR*OS)/OPHRS(I900))+
1020     & AINT((TOTHR*(1.-OS))/OPHRS(I900))
1030     REQMT(I900)=REQMT(I900)-TMRQT-NSECI(I900)

```

```

1040      GOTO 830
1050C
1060 830 SEUR(I900)=(USEHRS(I900)+NSECI(I900)*BSYHRS(I900)+TMHRS)/
1070      &      (NSECI(I900)*OPHRS(I900)+(REQMT(I900)+TMPQT)*OPHRS(I900))
1080C
1090C      COMPUTE SE ACQUISITION & OPERATIONS COST
1100 840 SEACQ=REQMT(I900)*CADB(I900)
1110      SEOPNS=USEHRS(I900)/OPHRS(I900)*CODB(I900)*PIUP
1120      SEFAC=FDB(I900)
1130      IF(TMRQT.GT.0) SEFAC=0
1140      SECOST(I900)=SEACQ+SEOPNS+SEFAC
1150      IF(ITYPE.GT.4.AND.ITYPE.LT.8) GO TO 850
1160      IF (ITYPE.EQ.9.AND.JTYPE.EQ.4) GOTO 850
1170      SECOST(I900)=SECOST(I900)/M
1180C      SAVE TOTAL COST FOR THE NETWORK SE COST ARC
1190 850 SECOST(I900)=AINT(SECOST(I900)+0.5)
1200      IF(IND.EQ.0) GO TO 880
1210      IF(LINES.LT.55) GO TO 855
1220      IPAGE=IPAGE+1
1230      WRITE(6,801) (SYSNAM(ISUB),ISUB=1,3),DATE,IPAGE
1240      WRITE(6,802) CHARS(5),CHARS(6)
1250      WRITE(6,803) PIUP
1260      WRITE(6,804)
1270      LINES=7
1280 855 CHOURS=0.
1290      IF(ITYPE.EQ.1.OR.ITYPE.EQ.5.OR.JTYPE.EQ.1)
1300      & CHOURS=NSECI(I900)*BSYHRS(I900)
1310      REQMT(I900)=REQMT(I900)+TMRQT
1320      USEHRS(I900)=USEHRS(I900)+TMHRS
1330      SEACQ=REQMT(I900)*CADB(I900)
1340      SEOPNS=(USEHRS(I900)/OPHRS(I900))*CODB(I900)*PIUP
1350      SEFAC=FDB(I900)
1360      TSECST(I900)=TSECST(I900)+SECOST(I900)
1370      TSECST(I900)=AINT(TSECST(I900)+0.5)
1380
1390      WRITE(6,860) SECODE(I900),(XSE(I900,ISUB),ISUB=1,3),NSECI(I900),
1400      & REQMT(I900),CHOURS,USEHRS(I900),SEUR(I900),SEACQ,SEOPNS,
1410      & SEFAC,TSECST(I900)
1420 860 FORMAT(' ',3X,I5,3X,3A4,3X,I4,4X,I4,2X,2F7.1,2PF6.1,OP4F11.0)
1430      LINES=LINES+1
1440      IF(I900.LE.NDSE) GO TO 880
1450      IF(I900.GT.NDSE) GOTO 875
1460      WRITE(6,870)
1470 870 FORMAT(' BASE')
1480      LINES=LINES+1
1490      GOTO 880
1500 875 IF(I900.NE.NDSE+NBSE) GOTO 880
1510      WRITE(6,877)
1520 877 FORMAT(' CTRF')
1530      LINES=LINES+1
1540C

```

```
1550 880 IF(INO.EQ.1) GO TO 900
1560      SEPTR=SEARCP(I900)
1570      CAP(SEPTR)=SECOST(I900)
1580      SECOST(I900)=SAVCST
1590 900 CONTINUE
1600      DO 1 I=1,NUMSER
1610      REOMT(25)=REOMT(25)+REOMT(I)
1620 1    CONTINUE
1630      RETURN
1640      END
```

★

Bibliography

1. Aerospace Studies Institute, Evaluation of Research and Development and Procurement Support of Operations in Southeast Asia (1 Jan 1965 - 31 Mar 1968)(U), Corona Harvest Report, Air University, Maxwell AFB, Alabama, Dec 1970.
2. AFALC/XRS. Network Repair Level Analysis Model Programmers Guide. Wright-Patterson AFB, Ohio. HQ AFLC, Nov 1981.
3. Armstrong, Lt Col Gerald R. Unpublished notes on "Max-flow Labeling Algorithm," Air Force Institute of Technology, Wright-Patterson AFB, Ohio, September 1982.
4. Barnes, J. Wesley and Paul A. Jensen. Network Flow Programming, New York: John Wiley and Sons Inc., 1980.
5. Bazaraa, Mokhar S. and John J. Jarvis. Linear Programming and Network Flows, New York: John Wiley and Sons Inc., 1977.
6. Bodnar, Al and James Callahan, Operations Research Analysts SDBE/B1-B SPO, ASD/AFSC. Personal Interview. Wright-Patterson AFB, Ohio, 6 July 1983.
7. Bradley, Gordon H., Gerald G. Brown, and Glenn W. Graves. "Design and Implementation of Large Scale Primal Transshipment Algorithms", Management Science, Vol.24, No.1, 1-34, September 1977.
8. Briskin, Larry and Terry Mitchell, Network Repair Level Analysis Models User's Guide, AFALD/XRS(AFLC), Wright-Patterson AFB, Ohio.
9. Briskin, Lawrence and Terry Mitchell, Operations Research Analysts AFALD/XRS, HQ AFLC. Personal Interview. Wright-Patterson AFB, Ohio, 27 June 1983.
10. Clark, Lt Col Thomas. "Policy Analysis Model for the Air Force Logistics System Report #1. Air Force Institute of Technology, Wright-Patterson AFB, Ohio,
11. Dept. of the Air Force. Integrated Logistic Support (ILS) Program. AFR 800-8. Wright-Patterson AFB, Ohio. HQ AFLC.
12. Dept. of the Air Force. Optimum Repair Level Analysis. AFLC Pamphlet 800-4. Wright-Patterson AFB, Ohio. HQ AFLC/AFSC, Oct 1970.

13. Dept. of the Air Force. Repair Level Analysis Procedures, AFLC/AFSC Pamphlet 800-4 (Final Draft). Wright-Patterson AFB, Ohio. HQ AFLC/AFSC, June 1983.
14. Dept. of the Air Force. Repair Level Analysis (RLA) Program. AFR 800-28. Wright-Patterson AFB, Ohio. HQ AFSC/AFLC, 29 May 1981.
15. Dessouky, Mohamed I. and Steve Phillips, Jr., "The Cut Search Algorithm With Arc Capacities and Lower Bounds", Management Science, Vol.25, No.4, 396-404 (April 1979).
16. Edmonds, Jack, and Richard M. Karp, "Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems", Journal of the Association for Computing Machinery Vol. 19, No. 2, 248-264 (April 1972).
17. Ford, L. R., Jr. and Fulkerson, D. R., Flows in Networks, Princeton, New Jersey: Princeton University Press, 1962.
18. Glover, F., J. Hultz, D. Klingman, and J. Stutz, "Generalized Networks: A Fundamental Computer-Based Planning Tool", Management Science, Vol.24, No.12, 1209-1220 (August 1978).
19. Horowitz, Ellis, and Sartaj Sahni, Fundamentals of Computer Algorithms. Potomac, Maryland: Computer Science Press, Inc., 1978.
20. James, J. H., and L. D. Servi, "The Use of Network Theory to Solve an ORLA Problem", Working Paper No. 21418. Bedford, Mass: The MITRE Corporation, Bedford Operations, September 1977.
21. Rhys, J. M. W., "A Selection Problem of Shared Fixed Costs and Network Flows", Management Science, Vol.17, No.3: 200-207 (November 1970).
22. Trempe, Maj Robert E., USAF, Squadron Leader Herbert E. Trichlin RAAF, and Lt Col Thomas Clark, "Complex Multi-Echelon Inventory System Management Using A Dynamic Simulation Model", Decision Sciences, Vol. 14, 389-407 (July 1983)

VITA

Captain Gary W. Arnett was born on 1 April, 1952 in Portsmouth, Virginia. He is married to the former Michelle Rodgers of Arlington, Virginia. They have two children, Becky and Brian, ages 4 and 1 1/2 respectively. He received a B.S. in Industrial Engineering and Operations Research from VPI in 1974. As a US Army Engineer Officer, he has served as a Combat Engineer Platoon Leader in the Federal Republic of Germany and as Chief of the Engineering and Management Analysis Branch of the Facility Engineer at Fort Belvoir, VA. Since his transfer to the US Air Force in November of 1980, he has served as an O.R. analyst in the ASD/AFLC Joint LCC Working Group at Wright-Patterson AFB until his acceptance at the Air Force Institute of Technology.

Permanent Address: 4713 Red Coat Road
Va. Beach, VA 23455

Captain Nicholas W. Reybrock was born on 16 July, 1948 in Plainville, Massachusetts. He is married to the former Kathleen Ann Marko of Cheyenne, Wyoming. They have one daughter, Kasey age twelve. He attended the University of Utah under the Airman Education Commissioning Program and graduated Magna cum Laude in 1976, with a B.S. in Chemistry. Captain Reybrock has 16 years in the Air Force. During his first 9 years he served as an electronics technician and as a chemistry laboratory technician at McClellan AFB CA, and as a gas chromatography technician at Eilison AFB AK. After receiving his commission in 1976, he has served as a Weapons Controller at a mobil TAC unit and a remote Alaskan radar site. He has also served as a Air Surveillance Officer and Radar Inputs and Countermeasures Officer at the 21st NORAD Region in New York. He was serving as a Operations Staff Officer and Standardization and Evaluation Officer at the 21st NORAD Region prior to his acceptance at the Air Force Institute of Technology.

Permanent Address : 106 No. Bedford Street
Georgetown DE 19947

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED		1b. RESTRICTIVE MARKINGS	
2a. SECURITY CLASSIFICATION AUTHORITY		3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited	
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE			
4. PERFORMING ORGANIZATION REPORT NUMBER(S) AFIT/TCR/OC/83D-3		5. MONITORING ORGANIZATION REPORT NUMBER(S)	
6a. NAME OF PERFORMING ORGANIZATION School of Engineering	6b. OFFICE SYMBOL (If applicable) AFIT/EN	7a. NAME OF MONITORING ORGANIZATION	
6c. ADDRESS (City, State and ZIP Code) Air Force Institute of Technology Wright-Patterson AFB, Ohio 45433		7b. ADDRESS (City, State and ZIP Code)	
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Concepts and Analysis Div	8b. OFFICE SYMBOL (If applicable) XRS/AFALC	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER	
8c. ADDRESS (City, State and ZIP Code) Bright-Patterson AFB, Ohio 45433		10. SOURCE OF FUNDING NOS.	
11. TITLE (Include Security Classification) see box 19		PROGRAM ELEMENT NO.	TASK NO.
		PROJECT NO.	WORK UNIT NO.
12. PERSONAL AUTHOR(S) Gary W. Arnett Captain, USAF Nicholas W. Reybrock Captain, USAF			
13a. TYPE OF REPORT Thesis	13b. TIME COVERED FROM TO	14. DATE OF REPORT (Yr., Mo., Day) December 1983	15. PAGE COUNT 151
16. SUPPLEMENTARY NOTATION			
<p>Approved for public release: LAW AFB 180-17, Lynn E. Wolaver Dean for Research and Professional Development Air Force Institute of Technology (AFIT) Wright-Patterson AFB OH 45433</p>			
17. COSATI CODES		18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)	
FIELD	GROUP	SUB. GR.	
19. ABSTRACT (Continue on reverse if necessary and identify by block number) Title : ENHANCEMENTS TO THE NETWORK REPAIR LEVEL ANALYSIS (NRLA) MODEL USING MARGINAL ANALYSIS TECHNIQUES AND CENTRALIZED INTERMEDIATE REPAIR FACILITY (CIRF) MAINTENANCE CONCEPTS Thesis Chairman : Gerald R. Armstrong, LtCol, USAF			
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED <input checked="" type="checkbox"/> SAME AS RPT. <input type="checkbox"/> DTIC USERS <input type="checkbox"/>		21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED	
22a. NAME OF RESPONSIBLE INDIVIDUAL Gerald R. Armstrong, LtCol, USAF		22b. TELEPHONE NUMBER (Include Area Code)	22c. OFFICE SYMBOL

Applied network theory and marginal analysis concepts were utilized to design, computerize, verify, and evaluate, three major software modifications to the Network Repair Level Analysis (NRLA) model. First, a preprocessor subroutine using marginal analysis techniques was developed and tested to reduce the computer processing requirements of the program. Second, a new network labeling algorithm which solve the max-flow min-cut problem is presented. This algorithm performs 100 times faster than the current algorithm and 73 times faster than the highly efficient, commercially available, primal networking code known as GNET. Third, for the first time a networking structure has been designed which allows for the inclusion of Centralized Intermediate Repair Facilities (CIRF) in the repair level analysis decision process.

These products greatly expand the NRLA model's capability while at the same time improving its operational efficiency. Through their integration and use, System Program Managers have a comprehensive analytical tool to effectively conduct repair level analysis and to design more cost-effective logistical structures to support the operation of Air Force systems.

The NRLA program is hosted on the CREATE Operating System and contains approximately 5500 lines of computer code. It consists of a main routine and twelve major subroutines. The results from the NRLA model are used by logistical planners to quantify the potential cost impacts associated with alternative maintenance plans. As the technological complexity of weapons systems has increased new and innovative logistical support systems are required to maximize the system's operational capability while minimizing life cycle costs. The above enhancements to the NRLA model were designed to meet these new challenges. This research effort was sponsored by the Concepts and Analysis Division, Air Force Acquisition Logistics Center (XRS/AFALC/AFLC).

END

FILMED

DANIC